DOCUMENT RESUME

ED 240 978 IR 010 969

AUTHOR	Streibel, Michael J., Comp.
TITLE	Proceedings of the Annual Penn State Microcomputer Information Exchange Conference (2nd, University Park, Pennsylvania, March 11-12, 1983).
PUB DATE	Mar 83
NOTE	250p.; Some sections may be marginally legible due to light or broken type.
PUB TYPE	Collected Works - Conference Proceedings (021) -- Reports - Descriptive (141)
EDRS PRICE	MF01/PC10 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; Computer Literacy; *Computer Oriented Programs; *Computer Programs; Computer Science Education; Databases; Fantasy; *Instructional Materials; *Microcomputers; Physical Environment; Problem Solving; Programing Languages; Staff Development
IDENTIFIERS	Computer Anxiety; *Computer Uses in Education; Interactive Systems; LOGO Programing Language

ABSTRACT

This collection of 17 conference presentations includes (1) "Project LOGO: A Study in Cognitive Enhancement Using Microcomputers," Henry Dobson; (2) "Tender Loving Care for Your Terrific Little Computer (TLC for your TLC)," Carol Dwyer and Karl Kelly; (3) "Teaching Micro-Literacy to Kids," Robert Gillingham; (5) "Teaching Music with the Apple II Microcomputer," William Higgins; (6) "The Design of a Microcomputer Data Base for Educators," Carol Jean Howard, Ronald Aitken, and Timothy Robinson; (7) "A Computer Programming Curriculum for High School (I, II, III), Newton Kulp; (8) "Microcomputers in Engineering Graphics," William H. Park; (9) "Computer Camps for Children and Adults in America," Joan Parrett; (10) "Parallel Microprocessors: Mainframe Power for Micro Prices," Douglas J. Parsons; (11) "Microcomputer Applications for Laboratory Sciences," Thomas Sandusky and Jeffrey D. Charney; (12) "Computer-Managed Instruction: Will I Control It or Will It Control Me?" Carol Scheftic; (13) "Computer Courseware for the Deaf and Learning Disabled: What's Being Done?," Casey Stone and Kathleen Smith; (14) "Microcomputer Anxiety," John Tongue; (15) "Computer Literacy for Professionals," Leslie E. Vance; (16) "Staff Development for Computer Literacy--A Model Way?" James P. West; and (17) "The Unexpected Treasure: Developing Problem Solving Skills through Interactive Fantasy," J. Alan Whiteside. The conference agenda is included, as well as lists of the planning committee and hosts. (LMM)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

- This document has been reproduced as received from the person or organization originating it.
- Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

ED240978

Proceedings of the Second Annual Penn State Microcomputer Information Exchange Conference

Michael J. Streibel, compiler

IR010969

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Michael J. Streibel
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

CONFERENCE PLANNING COMMITTEE

Chairmen:

Dr. Michael J. Streibel, Assistant Professor, Instructional Systems Program, College of Education

Dr. Chester M. Smith, Assistant Professor of Computer Science, College of Science

Dr. Robert F. Nicely, Jr., Coordinator of Continuing Education and Commonwealth Relations, College of Education

Ms. Melinda G. Harr, Conference Coordinator

CONFERENCE HOSTS

Instructional Systems Graduate Students

Ms. Barbara Bonham
Mr. Angel Caraballo-Rios
Ms. Banjert Chongapiratanakul
Mr. John Dobos
Ms. Mary Lynn Dreibelbis
Ms. Carol Dwyer
Ms. Suzanne Fish
Mr. Alvaro Galvis
Mr. John Glantz
Ms. Jeanne Howard
Mr. Karl Kelly
Mr. William Matten
Ms. Arlene Mitchell
Ms. Cheryl Myers
Ms. Joan Parrett
Ms. Kathy Smith
Ms. Casey Stone
Mr. John Tongue
Mr. Les Vance
Mr. James West
Mr. Alan Whiteside

TIME	TOPIC	SPEAKER	ROOM
FRIDAY, MARCH 11			
10:00 a.m. - 1:00 p.m.	FINAL REGISTRATION		Keller Lobby
10:00 a.m. - 4:45 p.m.	VENDOR EXHIBITS		Keller Lobby
1:00 - 1:15 p.m.	WELCOMING REMARKS	Dr. Michael J. Streibel Conference Co-Chairman Dr. Henry J. Hermanowicz Dean, College of Education Penn State Dr. Paul W. Welliver Professor, Instructional Systems Program Penn State	Keller Auditorium (first floor)
1:15 - 2:15 p.m.	OPENING GENERAL SESSION "Quality and Equity in Educational Computing: The School Consumers Responsibility"	Dr. P. Kenneth Komoski President and Executive Director EPIE Institute	
2:30 - 4:45 p.m.	CONCURRENT SESSIONS		
2:30 - 4:30 p.m.	COMPUTER LAB SESSION (by preregistration only) "Word Processing"	Dr. Thomas F. Barone	405 Keller
2:30 - 3:30 p.m.	SESSION I		
	PENN STATE SHOWCASE		
	"Use of Microcomputers in Nursing Instruction"	Mr. George H. McNeal	403 Keller
	EDUCATION PRESENTATIONS		
	"Project LOGO: A Study in Cognitive Enhancement"	Mr. Henry Dobson	312 Keller
	"Computer-Managed-Instruction (CMI): Will I Control It or Will It Control Me?"	Ms. Carol Scheftic	313 Keller
	"Teaching Music With the APPLE II Microcomputer"	Dr. William Higgins	314 Keller

TIME	TOPIC	SPEAKER	ROOM
2:30 - 3:30 p.m.	PROFESSIONAL PRESENTATION		
	"TLC for Your TLC (Tender Loving Care for Your Terrific Little Computer): Micro-Maintenance"	Ms. Carol Dwyer Mr. Karl Kelly Mr. Brad Weirich	402 Keller
	UNSCHEDULED USER DEMONSTRATIONS		401 Keller
	VENDOR DEMONSTRATION	Tandy Radio Shack	204 Keller
3:45 - 4:45 p.m.	SESSION II		
	DISCUSSION		
	Small Group Discussion of "Quality and Equity in Educational Computing"	Dr. P. Kenneth Komoski	402 Keller
	PENN STATE SHOWCASE		
	"Microcomputers in Engineering Graphics"	Dr. William H. Park	403 Keller
	EDUCATION PRESENTATIONS		
	"The Unexpected Treasure: Developing Problem-Solving Skills Through Interactive Fantasy"	Mr. Alan Whiteside	312 Keller
	"Microcomputer-Assisted Curriculum Management and IEP Development System"	Dr. John C. Flohr	313 Keller
	"Microcomputer Applications for High School Laboratory Sciences"	Dr. Thomas Sandusky	314 Keller
	UNSCHEDULED USER DEMONSTRATIONS		401 Keller
	VENDOR DEMONSTRATION	IBM	204 Keller

TIME	TOPIC	SPEAKER	ROOM
FRIDAY EVENING, MARCH 11			
7:00 - 9:00 p.m.	FILMS AND DISCUSSION Films 1. "Now the Chips Are Down" 2. "The Mind Machines" 3. "Don't Bother Me, I'm Learning"		Nittany Lion Inn Colonial Room I
7:00 - 9:00 p.m.	SIGCUE CASH BAR RECEPTION Special Interest Groups of Computer Using Educators		Nittany Lion Inn Colonial Room II
SATURDAY, MARCH 12			
8:30 a.m.	Coffee and Donuts Available		Third & fourth floor hallways in Keller
8:30 - 11:00 a.m.	VENDOR EXHIBITS		Keller Lobby
9:00 - 10:45 a.m.	CONCURRENT SESSIONS		
8:45 - 10:45 a.m.	COMPUTER LAB SESSION (by preregistration only) "Financial Analysis"	Dr. Thomas F. Barone	405 Keller
9:00 - 9:45 a.m.	SESSION I PENN STATE SHOWCASE "Microcomputers in Agriculture"	Dr. George Hussey	403 Keller

TIME	TOPIC	SPEAKER	ROOM
10:00 - 10:45 a.m.	PROFESSIONAL PRESENTATIONS		
	"Parallel Micro-processing: Getting Mainframe Power for Micro Prices"	Mr.Doug Parsons	305 Keller
	"Techniques for Evaluating Microcomputer Software"	Dr. Charles G. Forsythe Ms. Donna J. Niessner	310 Keller
	UNSCHEDULED USER DEMONSTRATIONS		401 Keller
	VENDOR DEMONSTRATIONS	Monroe Systems for Business Victor	204 Keller 311 Keller
11:00 a.m. - 12:00 noon	CLOSING GENERAL SESSION	Dr. Chester M. Smith Conference Co-Chairman	Keller Auditorium
	"Planning Strategies for the Use of Microcomputers in Education"	Mr. Gary Augustson Director of Computer and Information Systems Penn State	

PLEASE NOTE: THERE WILL BE A MESSAGE BOARD PLACED IN THE LOBBY FOR THOSE RECEIVING MESSAGES DURING THE CONFERENCE

Project LOGO: A Study in Cognitive Enhancement Using Microcomputers

Henry D. Dobson
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

PROJECT LOGO: A STUDY IN COGNITIVE ENHANCEMENT USING MICROCOMPUTERS

By Henry D. Dobson
Central Columbia School District
Bloomsburg, PA

c. 1983

For: The Second Annual Microcomputer Information Exchange Conference, Pennsylvania State University

Piaget once said, "There is no learning without experience." In the last ten years there have been shallow attempts to use computers to provide learners with these experiences in the form of drill and practices and / or simulations. However, these primitive approaches in computer usage have been strictly pragmatic and have required simple responses from the learner.

Even though the responses are sometimes accompanied with bells and whistles, these techniques still guide the student to the correct answers or down the "right" path. Using these techniques, and only these strategies, the micro's full potential will certainly be diminished as an educational tool of the 21st century. With the advances made daily in high technology, the role of the micro would remain at the QWERTY level.

The QWERTY phenomenon is based on the top row of keys on the standard typewriter. When the typewriter was first invented, the placement of the alphabet on the keys was selected to minimize key collisions by separating those keys that would most frequently follow one another. As technology advanced, improvements resolved the jamming problem but the QWERTY syndrome still exists.

This same type of phenomenon would exist with our usage of the microcomputer should we allow it. The yes/no drill and practice and simulations do have their place in the computer field, but not at the top. As our theoretical and technological knowledge increases daily, we can't, as educators, allow such low level techniques to become entrenched in our educational system. It would be the same as buying a sophisticated 35mm single lens reflex camera with all of its attendant sophistication, and then set the camera to the automatic position to take average pictures. Consider the

children who will be entering kindergarten in 1987, only four years from now, they will be the class os 2000. Will this average use of computers prepare them for life?

How many times do we as science educators ask our class to think of a solution or process in solving particular problems? Some modern day epistemologists believe that the key role of the computer, or"brain multiplier", might lie in its ability to enhance, nurture, and develop the thinking processes, but not replace it.

Just as Howe, Watt, and Whitney changed the world during the Industrial Revolution, a quiet revolution, a computer revolution has been started by Seymour Papert. Dr. Papert and his staff members at the artificial intelligence laboratory at MIT have developed a philosophy of education called LOGO. Using many of the theories developed by Piaget, Papert expanded the image of the child and how children learn to think, a type of learning that comes naturally, gives meaning, and must be fun. LOGO allows children to explore their environment in which drawings, language, and mathematics can be learned in a natural way just as a child learns to speak or walk. It is a way of using the computer where there is "no threshold and no ceiling." To this rich and joyful environment, a type of learning and play that go hand in hand, LOGO lets children begin to realize the most powerful idea of all: that each of these children have the capability of creating on a screen very powerful ideas.

In using this approach, these programmers become "toolmakers". Each newly created idea allows for the creation of more powerful ideas. LOGO allows the programmer to speak to the computer extending the natural process of communication. It's like learning German in Germany, not from a textbook. Therefore, it is vital that children be exposed to a computer language that would have a lifelong effect

on the way they think. This cognitive engineering would incorporate the natural growth process of the mind's structure and organization in an environment that is closely associated with the real world of "learn by doing". This new educational resource has enormous potential for developing the social, aesthetic, emotional, and intellectual abilities of these 21st century learners.

LOGO is a child centered and, on the surface, unstructured and non-hierarchical but is bound by parameters well-defined within the highly structured LOGO environment. These interesting environments are the "seed beds" of learning problem solving strategies. With the work of Polya, Newell, and Simon these strategies necessitate the development of the cognitive abilities that would include genegalization, planning, abstraction, and foremost, understanding. Research has indicated that doing projects in these areas could develop the intellectual structures that would nurture this aspect of the cognitive domain.

Dr. Papert, in developing LOGO, has created a micro-learning environment that is resonant with the child's own real world, a syntonic learning experience. Each of these experiences can then be focused and expanded to become meaningful to the child to a greater extent that would be possible by any cadre of educators. For the child in this microworld environment, it is a learning without being taught. The educator can now accept the higher level role of clarifying, inspiring, guiding, and motivating the child as he become the builder of his own intellectual structures.

We as science educators want citizens who will be able to cope with the challenges of a high technology society of the present, and moreso with the immediate future--a society so complex that we

need "thinkers" to permeate the computer culture of the not too distant future.

It is with this mission of creating a society of thinkers that Piaget, in his book, "The Science of Education and the Psychology of the Child", comments on the specific role of modern education. "If we desire to form individuals capable of inventive thought and of helping the society of tomorrow to achieve progress then it is clear that an education which is an active discovery of reality is superior to one which consists merely in providing the young with ready-made will to will with and ready-made truths to know with."

****REFERENCES****

Lawler, R. W. "Designing Computer-Based Microworlds," Byte, August 1982 138-160

Papert, Seymour, Mindstorms: Children, Computers, and Powerful Ideas. New York: Basic Books Inc., 1980

Piaget, Jean, The Science of Education and the Psychology of the Child. New York: Orion, 1970

---------- "Development and Learning", Journal of Research in Science Teaching. 1964, 176-186

Sund, Robert B. Piaget for Educators. Columbus: Merrill, 1976

Taylor, R. P. The Computer in the School: Tutor, Tool, Tutee. New York: Teachers College Press, 1980

Watt, D. H. "LOGO": A computer language as a learning environment. 99'er Magazine, May/June 1981, 58,59

SUMMARY OF THE LOGO PROJECT AT THE
CENTRAL COLUMBIA SCHOOL DISTRICT

The Central Columbia School District has instituted a LOGO program at the fifth and sixth grade levels. This program started in February and will continue through May of 1983.

It consists of two forty-five minute classes per day, three days per week. These classes are in addition to those regularly scheduled within the lower middle school curriculum. The current population of the fifth & sixth grades is about 420 students or 28 subjects per section.

The first objective of the LOGO project is to have students create their own learning environments, in which there is no fear of making mistakes; a place that is closely associated with their natural surroundings, an environment which allows the students to be active participants in their learning experience, thereby learning by doing. Students will create graphics programs that will help them learn to think in a logical manner. Using this technique, difficult concepts and ideas become easily understood as realities on the computer screen.

The second objective in introducing LOGO is to have the students develop skills in problem-solving procedures. A problem can be divided into its components and a separate procedure written for each of these components. These procedures are used as building blocks to further develop other procedures. Students will be able to create graphical procedures to develop basic geometric figures and then use them to create more complex configurations. Using the electronic sketchpad, students will be able to design spectacular shapes from very primitive procedural commands.

Activities related to objectives were selected from the LOGO miniworld. In order to achieve the first objective, turtle graphics was selected as the activity. The "turtle" is a small triangular shape that glows on the screen and like an arrowhead can be aimed in any direction. With this a student can draw a straight or curved line using a set of simple, primitive commands. As the student moves the "turtle" a trace remains on the screen. Students quickly proceed to drawing more complex figures as they learn to combine the many previously-learned geometric forms.

To satisfy the second objective, students will be introduced to trace out the pattern generated by the procedures.

As a means of documenting the results of the activities, students will be assessed for their cognitive level prior to the LOGO activities and then again at the completion of the activities. In order to control outside factors, a parallel control group will also be assessed during the treatment time period. The pre and post instrument used in the evaluation of the students' cognitive level was taken from Robert B. Sund's book, "Piaget for Educators: A Multi-Media Approach," 1976. Statistical techniques will be used to determine any gain in the group means as a result of the LOGO activities.

Tender Loving Care for Your Terrific Little Computer (TLC for your TLC)

Ms. Carol Dwyer & Mr. Karl Kelly
Instructional Systems Graduate Students
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

TLC FOR YOUR TLC
(or Tender Loving Care for your Terrific Little Computer)

Carol Dwyer and Karl Kelly
Division of Curriculum and Instruction
The Pennsylvania State University

A good philosophy relative to computer use is to work to prevent problems from occurring rather than spending hours trying to determine what caused a problem. Therefore, we would like to present information on how to avoid the problems which result from these sources:

physical environment - conditions around your computer (the air, the work area, equipment placement)

power supply - the electrical circuits, the cords, power controllers

electromagnetic interference - static electricity and interference from other equipment which have magnetic fields

operator error - any problem caused directly by user action or inaction

To maintain the proper environment for your computer:

1. Don't stack computer components unless thay are designed for stacking.

2. Don't place your computer near a heat source, and don't block ventilation openings in the components.

3. Don't eat or drink anything while working with your computer since this could result in gummy, clogged keys or shorted circuits.

4. Don't smoke in the vicinity of your computer.

5. Don't do dust-forming work near a computer because dusty innards and malfunctioning disk files will result.

6. Don't put magnets near your computer or any device which has magnetic parts (such as a radio with a solenoid cell or a telephone).

Do the following for your computer:

1. Arrange computer components conveniently. Your work area should have enough light and

space and be at the proper height.

2. Keep your documentation, software, and files organized.

3. Buy or make (and use) dust covers.

4. Place your printer on a separate table because over time vibrations will harm computer components.

5. Use a clamp to hold data lines securely to the computer so that if a component is moved, stress is not placed on the computer card inside.

6. Keep and operate your computer in an environment comfortable for people; keep both temperature and humidity at normal levels (60 to 80 degrees, 50% humidity), thus keeping static-forming conditions to a minimum.

7. Use a dedicated power line directly connected to the fuse box. A line of this type is like those installed for home freezers and electric stoves. If you don't follow this suggestion, be aware that power surges or dips can alter the form of your data.

Keep in mind that your computer is an electromagnetic device and is affected directly by alterations in power supply, static electricity, and magnetic fields. Do the following:

1. Place an antistatic mat under your work area or use an antistatic spray once a month in the area. Don't place your computer in an area with nylon carpeting.

2. Cords should not be unnecessarily close to individual components or each other in order to avoid magnetic interference. A TV used as a monitor should be a distance from the computer.

3. Find a way to discharge static electricity from your body. A good method is to ground the table on which your computer rests and then to touch the ground prior to using your computer. Another good method is to make it a habit to touch the on-off switch each time you return to your computer after walking away.

4. When touching the inside of the computer, keep the computer plugged into the wall outlet and the power off; before touching the boards, touch the power supply to discharge static electricity from your body.

A few things you can do to avoid sudden alterations in power supply:

1. Power cords should not be any longer than necessary or under chairs or in walkways in order to avoid accidentally pulling the plug.

2. Be aware of lights that dim or flash when a large motor starts or stops. If this happens, you must place a line conditioner on the power line (if the circuit is not already dedicated) or you must keep the device out of use when you are using your computer.

3. If the printer is plugged into the same outlet as your computer don't turn the printer on or off while the computer is reading from or writing to disk.

4. Check all connections periodically.

5. To avoid excessive accumulation of conductive material, dust the vents of your computer as needed by using inert gas under pressure and blowing dust from the inside to outside, or dust gently from the outside using a vacuum cleaner.

The ideal environmental conditions for your computer have been explained in detail. It is generally quite stable. However, your floppy disk can float around and get into awful messes. To protect your disks:

1. Don't bend or fold a disk. Don't set anything on top of a disk.

2. Don't touch exposed portions of a disk and don't smoke while handling a disk. These contaminants are very large in comparison to the space which is used by "bits" of data.

3. Don't expose the disk to liquids or vapors.

4. Label and date all your disks and keep them in holders that keep them clean. Labeling and proper storage helps you find what you need without spending time inserting disks, booting up, and reading disk directory contents.

5. Don't place disks near large motors or magnetic fields. Don't place any metal object, even screw drivers or paper clips which may have become magnetized, close to a disk.

6. Don't write directly on the disk cover unless you use a felt-tip pen. You should write on the sleeve with the disk removed or on labels which you can press on the cover. Don't make

dust by erasing the labels. An excellent method is to number the disk and the corresponding jacket; then do all writing and label changes on the jacket.

7. Don't turn the computer off while the disk drive is in use. Wait until the computer finishes using the disk; if necessary, reset the computer if it is stuck in a loop.

8. Use the proper type of disk for the drive, and insert the disk correctly.

9. Don't let the door of the disk drive snap shut since it could harm the disk.

10. Don't let a disk become too hot or too cold. If it has been exposed to a minor difference in temperature, allow a few minutes for it to return to room temperature before you use it. If exposed to extreme temperatures (i.e., outside 55 to 100 degrees), allow 24 hours before using it.

11. Don't use anything except top-quality disks which are fully tested and certified.

12. Even though you use good disks, don't expect a disk to last forever. Estimates of the active life of a disk range from 40 hours to 240 hours. Have copies of important programs and data.

13. Don't attempt to use both sides of a one-sided disk. The second side probably failed certification when produced and for that reason the particular disk was classified as one-sided. Also, a drive which is designed to handle one-sided disks may harm the side of the disk which is not in use.

Some precautions related directly to your disk drive are:

1. Before each use inspect the exposed surface of the disk for scratches, folds, or wide shiny rings, and check the center hole for wear. These are signals that adjustments are needed.

2. Clean disk drive heads periodically, preferably once per month since metal oxide can accumulate. A recommended procedure is to remove the disk drive cover and clean the head with a lint-free swab dampened with isopropyl alcohol. Head-cleaning kits are also available in the form of a special disk which you place in the disk drive, and while the disk drive operates and the disk spins

for a few seconds, the disk head is cleaned by the cleaner on the disk.

3. Periodically check and, if necessary, adjust disk drive speed and the alignment of read/write head over tracks. Special programs are available to direct you in these processes.

If after you have taken the preceding precautions, you still persist in making human errors, you can still have problems. Some of these mistakes are:

1. using the wrong commands or incompatible software,

2. accidentally pulling the cord from the wall outlet,

3. inserting integrated circuit boards without checking whether the computer is off,

4. using mismatched peripherals or incorrect interface cards,

5. making improper connections,

6. mishandling equipment such as drives or the printer, and

7. not saving files you think you saved.

Operator errors can be prevented by following a little maxim: Be certain you know what you're doing!

REFERENCES:

Leventhal, Lance A. & Irvin Stafford. Why do you need a personal computer? New York: John Wiley & Sons, Inc., 1981.

Zaks, Rodney. Don't (or how to care for your computer). Berkeley: Sybex, 1981.

YOUR COMPUTER ENVIRONMENT

To maintain proper environment for your computer:

1. Don't stack computer components unless they are designed for stacking.

2. Don't place your computer near a heat source, and don't block ventilation openings in the components.

3. Don't eat or drink anything while working with your computer since this could result in gummy, clogged keys or shorted circuits.

4. Don't smoke in the vicinity of your computer.

5. Don't do dust-forming work near a computer because dusty innards and malfunctioning disk files will result.

6. Don't put magnets near your computer or any device which has magnetic parts (such as a radio with a solenoid cell or a telephone).

Do the following for your computer:

1. Arrange computer components conveniently. Your work area should have enough light and space and be at the proper height.

2. Place your printer on a separate table because over time vibrations will harm computer components.

3. Keep your documentation, software, and files organized.

4. Buy or make (and use) dust covers.

5. Keep and operate your computer in an environment comfortable for people; keep both temperature and humidity at normal levels (60 to 80 degrees, 50% humidity), thus keeping static-forming conditions to a minimum.

6. Use a dedicated power line directly connected to the fuse box. A line of this type is like those installed for home freezers and electric stoves. If you don't follow this suggestion, be aware that power surges or dips can alter the form of your date.

To avoid electromagnetic interference:

1. Place an antistatic mat under your work area or use an antistatic spray once a month in the area.

2. Cords should not be unnecessarily close to individual components or each other in order to avoid magnetic interference. A TV used as a monitor should be a distance from the computer.

3. Find a way to discharge static electricity from your body. A good method is to ground the table on which your computer rests and then to touch the ground prior to using your computer. When you walk away from and then back to your computer, make it a habit to touch the on-off switch before you set down to work again.

4. When touching the inside of the computer, keep the computer plugged into the wall outlet, and before touching the boards, touch the power supply to maintain grounding of your body.

To avoid sudden alterations in power supply:

1. Power cords should not be any longer than necessary to avoid accidentally pulling the plug.

2. Be aware of lights that dim or flash when a large motor starts or stops. If this happens, you must place a line conditioner on the power line (if the circuit is not already dedicated) or you must keep the device out of use when you are using your computer.

3. Don't turn the printer on or off while the computer is reading from or writing to disk.

4. Check all connections periodically.

5. Dust the vents of your computer as needed by using inert gas under pressure and blowing dust from the inside to outside, or dust gently from the outside using a vacuum cleaner.

DISKS - HELP BEFORE HELP IS NEEDED

1. Don't bend or fold a disk. Don't set anything on top of a disk.
2. Don't touch exposed portions of a disk and don't smoke while handling a disk.
3. Don't expose the disk to liquids or vapors.
4. Label and date all your disks and keep them in holders that keep them clean.
5. Don't place disks near large motors or magnetic fields. Don't place any metal object, even screw drivers or paper clips which may have become magnetized, close to a disk.
6. Don't write directly on the disk cover. Write on the sleeve with the disk removed or on labels which you can press on the cover.
7. Don't let the door of the disk drive snap shut since it could harm the disk.
8. Don't turn the computer off while the disk drive is in use. Wait until the computer finishes using the disk; if necessary, reset the computer if it is stuck in a loop.
9. Don't let a disk become too hot or too cold.
10. Don't use anything except top-quality disks which are fully tested and certified.
11. Don't expect a disk to last forever. Estimates of the active life of a disk range from 40 hours to 240 hours. Have copies of important programs and data.
12. Don't attempt to use both sides of a one-sided disk.
13. Use the proper type of disk for the drive.
14. Before each use inspect the exposed surface of the disk for scratches, folds, or wide shiny rings, and check the center hole.
15. Insert the disk correctly.
16. Clean disk drive heads periodically, preferably once per month.
17. Adjust disk drive speed and alignment of read/write head over tracks.

Teaching Micro-Literacy to Kids

Dr. Robert Gillingham
Fayette Campus
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

"TEACHING MICRO-LITERACY TO KIDS"

DR. ROBERT GILLINGHAM, Penn State University, Fayette Campus

2nd Annual Microcomputer Information Exchange Conference
March 12, 1983 Pennsylvania State University, University Park

I wish to suggest an instructional model for teaching computer literacy to public school children, as early as the second grade, and who have no, or little, prior computer experiences.

***Keyboard manipulation and understanding the CRT screen format

***Understanding of good computing habits

***Methods of input

***Methods of output

***Methods of process

***Graphics and sound

The computer language used ought to be BASIC, and the students should be able to see various capabilities of the micro--how it is a counting machine, a game machine, a business machine, an educational tool.

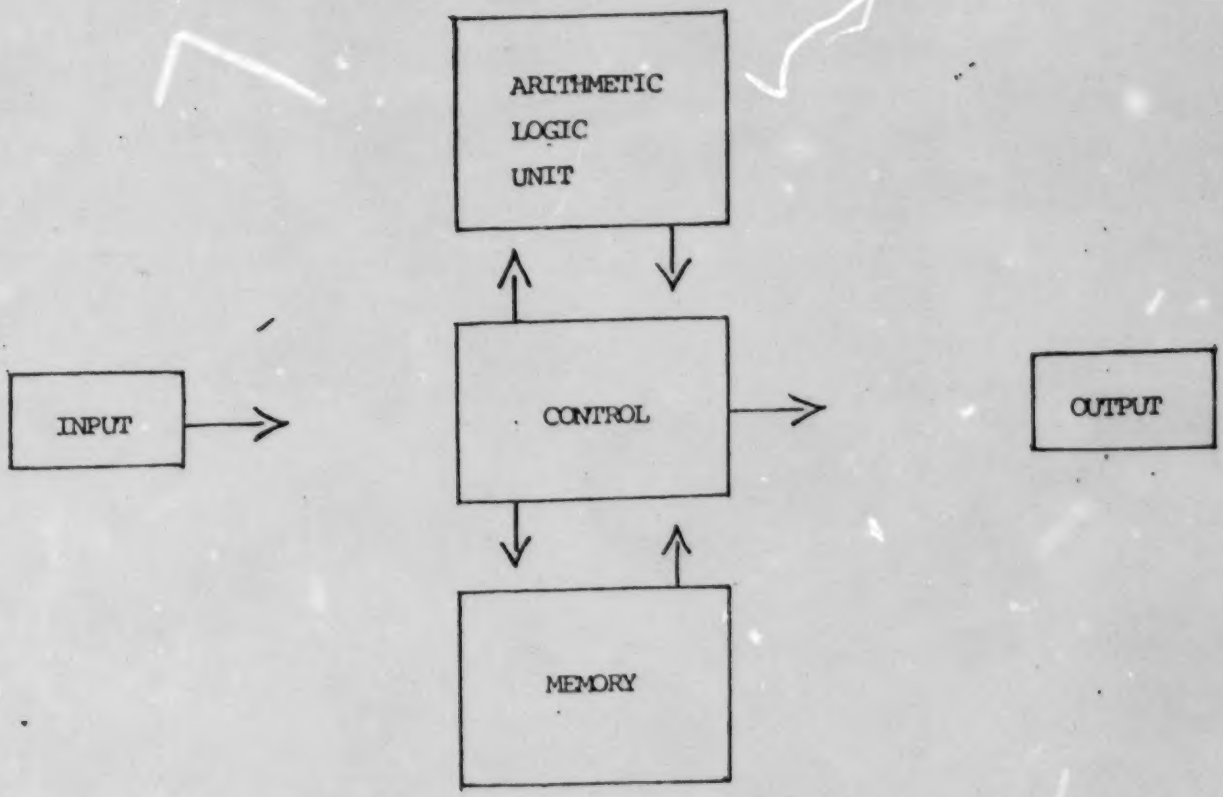
The above concepts have been regularly tested through a 15 hour, non-credit course I offer at Penn State, Fayette Campus, entitled "Kids & Komputers." At the end of the course, the children not only understand and use elementary computer science vocabulary, but sit at ease before the micro and use it as directed.

PROGRAM LISTING

```
100  PRINT "THIS PROGRAM DRAWS A RECTANGLE"
110  PRINT "TYPE IN HEIGHT, WIDTH"
120  INPUT H,W
130  FOR I=1 TO W
140  PRINT "-";
150  NEXT I
160  PRINT
170  FOR I=1 TO H-2
180  PRINT "*";TAB(W-1);"*"
190  NEXT I
200  FOR I=1 TO W
210  PRINT "-";
220  NEXT I
230  END
```

Susan D. Lipscomb, and Margaret Zuanich, BASIC Fun (New York: Avon, 1982), p. 123.

HOW A COMPUTER WORKS --




```
10  REM "ADDING 2 NUMBERS."
20  LET A=2  --
30  LET B=3
40  S=A+B
50  PRINT S
60  END
```

KIDS & KOMPUTERS

Penn State University, Fayette Campus
Dr. Robert Gillingham
Copyright 1982, by Dr. Robert G. Gillingham

KIDS & KOMPUTERS

Agenda— Day 1

Objective: To enable the operation of the CRT with reasonable confidence.

I. How To Power Up

II. Fundamentals of the CRT Keyboard

III. Elementary Program Commands in the BASIC Language

REM, PRINT LPRINT EMPTY PRINT, READ/DATA
LET, GO TO, END

IV. Elementary System Commands

CLS, SYSTEM "DIR", LOAD, SAVE, LIST, RUN, NEW, KILL, BREAK

V. Practice, Practice, Practice.

Some Vocabulary We Ought to Know:

1. hardware, software
2. CRT
3. Power Up (Boot Up)
4. Cursor
5. Statement
6. Program
7. Interactive
8. Input, Output
9. BASIC
10. COBOL
11. FORTRAN
12. Infinite Loop

SOME PROGRAMS WE WILL USE ON DAY 1.

```
10  REM "This Program Is An Infinite Loop"
20  N=0
30  A=2
40  N=N+A
50  PRINT N;
60  GOTO 40
70  END
```

```
10  REM "This Program Uses LET"
20  LET A=5
30  LET B=8
40  N=A+B
50  PRINT N
60  END
```

```
10  REM "This Program Uses READ/DATA"
20  READ A, B
30  DATA 5, 8
40  PRINT N=A+B
50  END
```

```
10  REM "This Program Can Count From 1 to 100 Faster Than"
11  REM "You Can"
20  LET N=1
30  PRINT N;
40  IF N > 1000 THEN 70
50  LET N=N+1
60  GOTO 30
70  END
```

QUESTION: What Command Does PRINT Become If You Want Output on the Line-Printer?

EXTRA FUN!!

Write a program that adds, subtracts, or multiplies two or three numbers. You could use either the LET statement or the READ/DATA statement.

-OR-

Write a program that gives the average age of the members of your family. The program could be alot like the READ/DATA example we saw in class. The data could be the ages of each family member. What you might need to learn is how to take an average.

HINT: You first add the ages of each family member and divide that sum by the total number of family members.

Example: 3+5+31+32=71 71 divided by 4=17.3

A computer needs to have all that written as

AGE = (3+5+31+32)/4

Penn State University, Fayette Campus

Dr. Robert Gillingham

Copyright 1982, by Dr. Robert G. Gillingham

KIDS & KOMPUTERS

Agenda--Day 2

Objectives: To provide further practice on the CRT

To promote an understanding of the interactive nature of the CRT

To introduce graphics by means of the PRINT and PRINT @ statements.

I. Enter and Run homework assignment of program to average the ages of family members.

II. Learn to use the various PRINT statements

III. More Practice on the CRT by playing an ESP Game

and

IV. Learn about the interactive nature of the CRT

Some programs we will work with:

```
10  REM "THIS PROGRAM WILL PUT MY NAME IN LIGHTS."
20  CLS
30  PRINT "MY NAME"
40  GOTO 20
50  END
```

```
10  REM
20  PRINT
30  PRINT "HUMANS CAN MAKE MISTAKES."
40  PRINT
50  PRINT "BUT TO REALLY GOOF THINGS UP--"
60  PRINT
70  PRINT "YOU NEED A COMPUTER."
80  END
```

```
10  REM
20  CLS
30  PRINT @300, "MY NAME"
40  GOTO 20
50  END
```

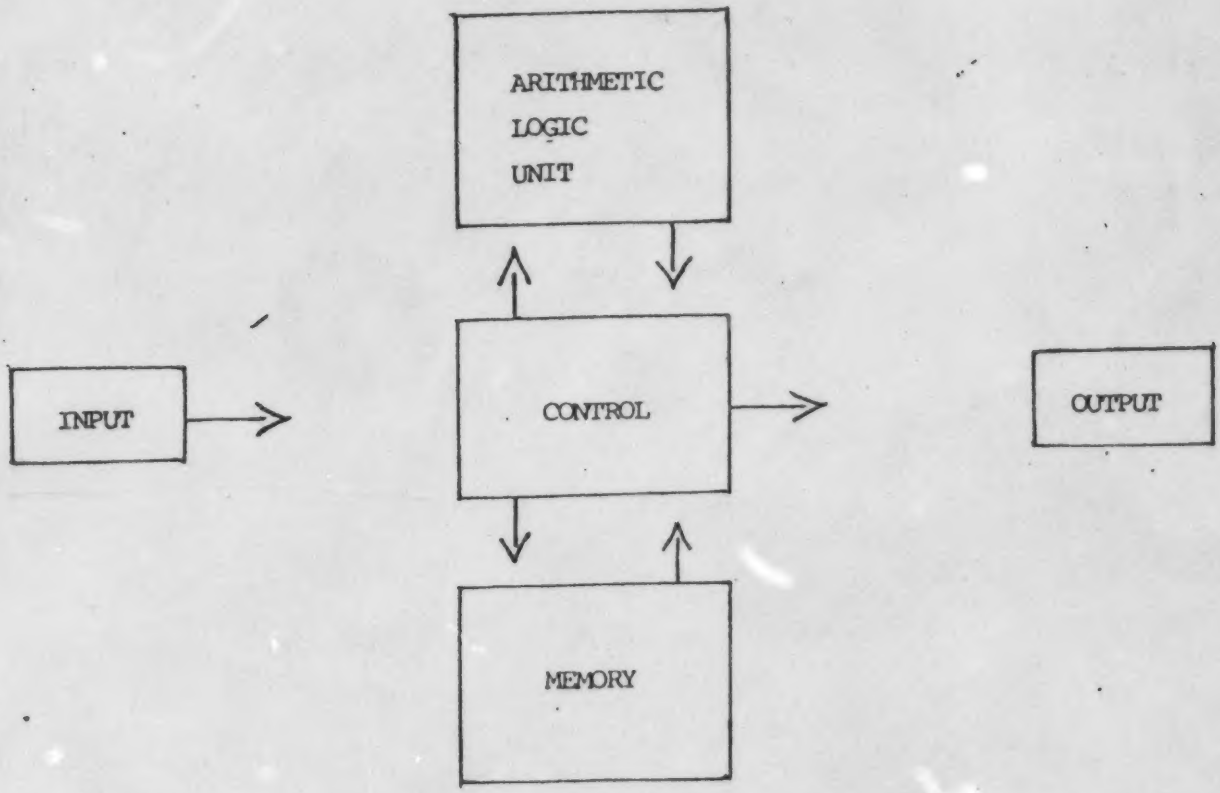
Ideas for the first and third programs above were originated from Bob Albrecht, et. al., TRS-80 Level II BASIC (New York: John Wiley, 1980), Chapter 3, pp. 39-74.

FUN HOMEWORK ASSIGNMENT:

Write a program in which you interact with the computer. It can be a lot like the one listed below.

```
10  REM "THIS PROGRAM ASKS ME MY NAME."
20  PRINT "WHAT IS YOUR NAME?"
30  PRINT
40  INPUT A$
50  PRINT "HELLO, ";  A$
60  END
```

HOW A COMPUTER WORKS --




```
10  REM "ADDING 2 NUMBERS."
20  LET A=2  --
30  LET B=3
40  S=A+B
50  PRINT S
60  END
```

Penn State University, Fayette Campus
Dr. Robert Gillingham
Copyright 1982, by Dr. Robert G. Gillingham

KIDS & KOMPUTERS

Agenda--Day 3

Objective: To initiate the participants to elementary concepts of graphics.

I. Review the INPUT statement by entering and running homework assignment. Demonstrate changes to ESP game.

II. Graphics: A). Demonstration and discussion of Program "ME"
1. Use of PRINT

B). Demonstration and discussion of Program "FLEET"
1. Use of TAB

C). Hands-on experience with Program "WOODSTOCK"

III. Use of Line-Printer.

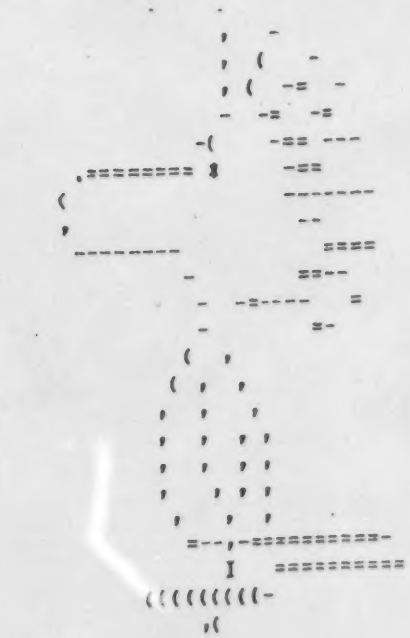
HOMEWORK ASSIGNMENT: Write a program in which you draw a map of your favorite state. The program could be a lot like the one shown in the handout, or you could just outline the state's form and put an X where the capitol is located.

J.D. Robertson, and John P. Grillo, TRS-80 Graphics
(Dubuque: Iowa, Wm. C. Brown, 1981), p. 5.

Problem 1.4 Sketch a picture of Woodstock.

Solution

```
10 'filename:"sk4"
20 ' purpose: draw Woodstock
30 '  author: jdr 8/80
40 '
50 LPRINT "            ,   -"
60 LPRINT "            , (   -"
70 LPRINT "            , (  -=  -"
80 LPRINT "            -  -=  -="
90 LPRINT "          -(     -== ---"
100 LPRINT " ,======== $      -=="
110 LPRINT "(                  -------"
120 LPRINT ",                   --"
130 LPRINT " --------             ===="
140 LPRINT "         -           ==--"
150 LPRINT "          -  -=----   =="
160 LPRINT "          -           =-"
170 LPRINT "        (   ,"
180 LPRINT "       ( ,  ,"
190 LPRINT "      ,  ,    ,"
200 LPRINT "      ,  ,  , ,"
210 LPRINT "      ,  ,  , ,"
220 LPRINT "      ,    , , ,"
230 LPRINT "       ,   ,  ,"
240 LPRINT "        =--,-----------"
250 LPRINT "           I   ==========="
260 LPRINT "      (((((((((-"
270 LPRINT "          ,("
280 END
```

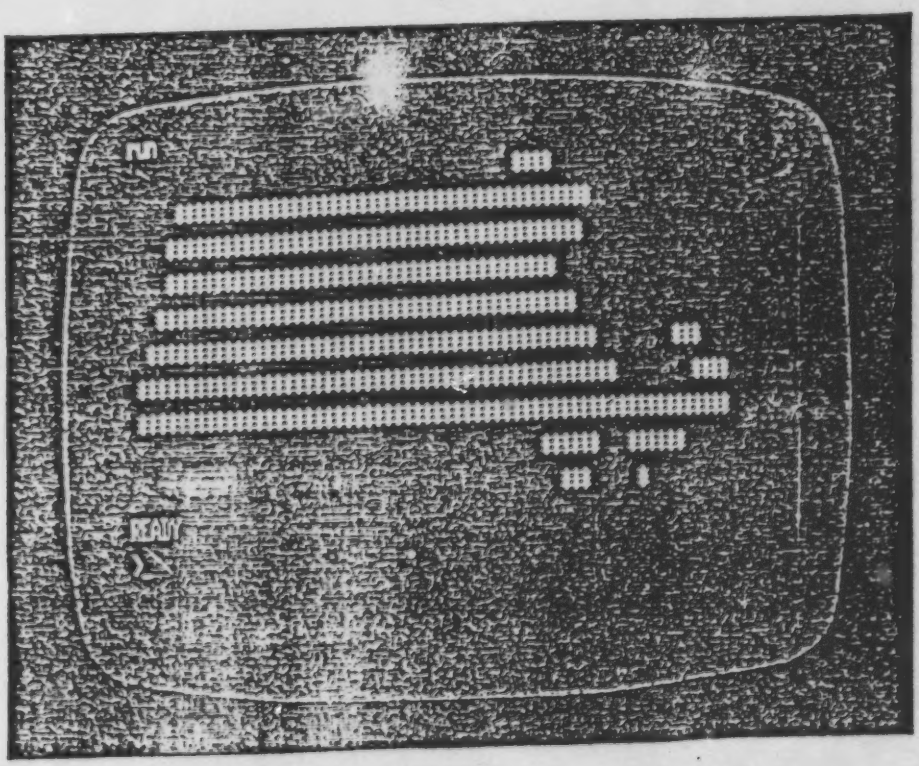


J.D. Robertson, and John P. Grillo, TRS-80 Graphics (Dubuque: Iowa, Wm. C. Brown, 1981), p. 3.

Problem 1.2 Draw a picture of the state of Massachusetts.

Solution

```
10 'filename:"sip2"
20 ' purpose: Massachusetts
30 '  author: jdr 8/80
40 '
50 PRINT "                                                  ****"
60 PRINT "      ********************************************"
70 PRINT "     *********************************************"
80 PRINT "     *********************************************"
90 PRINT "    **********************************************"
100 PRINT "  ***********************************************        ***"
110 PRINT " ************************************************        ****"
120 PRINT " ************************************************************"
130 PRINT "                                                  ******   ******"
140 PRINT "                                                   ***      *"
150 END
```



Penn State University, Fayette Campus
Dr. Robert Gillingham
Copyright 1982, by Dr. Robert G. Gillingham

KIDS & KOMPUTERS

Agenda—Day 4

Objectives: To reenforce knowledge of two kinds of graphics--line and pixel

To introduce Karel The Robot

I. Enter and Run homework assignment of map (line graphics). Participant uses the line printer.

II. Further work with line graphics. Participant designs and runs on line-printer own line graphics program.

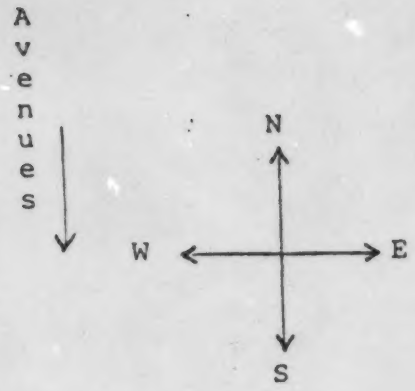
III. Demonstration of concept of pixel graphics. Participant runs pixel graphics program.

IV. Introduction to Karel The Robot. Understanding of its basic commands: move, turnleft, pickbeeper, putbeeper, turnoff

---

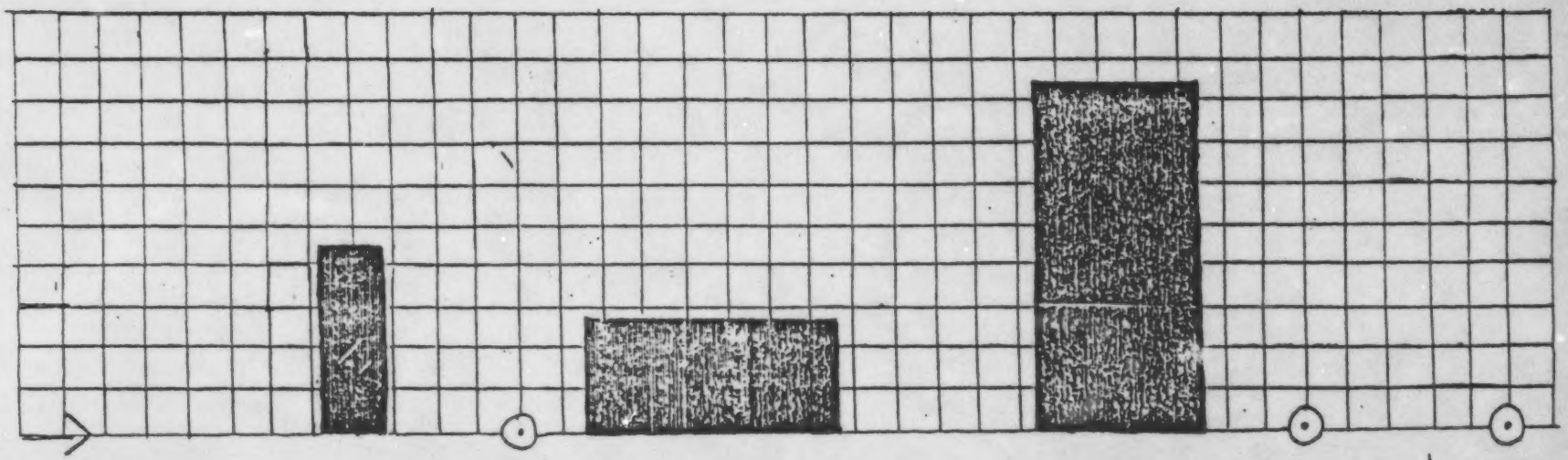
PLEASE NOTE: Our last session will be devoted to your developing and running a simple program of your own. In order to do this, you ought to know the elementary commands on handouts 1, 2, and 3. For example, you ought to know that BASIC programs have line numbers, statements, END, and so on. You also ought to come to class with an idea for your program. Some people might want to do an arithmetic problem (adding 3 numbers), some people might want to make a game (like ESP), and even others might want to do a graphics display.

KAREL THE ROBOT



Streets ←

Adapted from Richard Pattis, Karel The Robot (New York: John Wiley, 1981).



PROGRAM LISTING

```
100  PRINT "THIS PROGRAM DRAWS A RECTANGLE"
110  PRINT "TYPE IN HEIGHT, WIDTH"
120  INPUT H,W
130  FOR I=1 TO W
140  PRINT "-";
150  NEXT I
160  PRINT
170  FOR I=1 TO H-2
180  PRINT "*";TAB(W-1);"*"
190  NEXT I
200  FOR I=1 TO W
210  PRINT "-";
220  NEXT I
230  END
```

Susan D. Lipscomb, and Margaret Zuanich, BASIC Fun (New York: Avon, 1982), p. 123.

PROGRAM LISTING

```
100  PRINT "WATCH WHAT I CAN MAKE"
110  PRINT
120  PRINT
130  FOR K=1 TO 9
140  PRINT TAB(K);"HI";TAB(20-K);"HO"
150  NEXT K
160  PRINT TAB(10);"HI"
170  FOR K=11 TO 20
180  PRINT TAB(20-K);"HO";TAB(K);"HI"
190  NEXT K
200  END
```

Susand D. Lipscomb, and Margaret Zuanich, BASIC Fun (New York: Avon, 1982), p. 126.

F L E E T

PROGRAM LISTING

```
100  PRINT "WATCH THIS FLEET OF ROCKET SHIPS BLAST OFF!"
105  PRINT
106  PRINT
107  PRINT
108  PRINT
109  PRINT
110  LET R=INT(RND(1)*20+2)
120  PRINT TAB(R);"^";TAB(R+15);"^"
125  FOR I=1 TO 3
130  PRINT TAB(R-1);"I I";TAB(R+14);"I I"
135  NEXT I
140  PRINT TAB(R-2);"(---)";TAB(R+13);"(---)"
150  FOR J=1 TO 15
160  FOR I=1 TO 100
170  NEXT I
180  PRINT
190  NEXT J
200  GOTO 110
210  END
```

Susan D. Lipscomb, and Margaret Zuanich, BASIC FUN
(New York: Avon, 1982), p. 133.

Penn State University, Fayette Campus
Dr. Robert Gillingham
Copyright 1982, by Dr. Robert G. Gillingham

KIDS AND KOMPUTERS

Agenda—Day 5

Objective: To enable the participant to design, enter, and run his own computer program.

I. Individual Programs entered and run.

II. Demonstration of Sound and High Resolution Graphics.

III. "Award" Program.

Dear Computing Student,

I hope that you have enjoyed these five days (fifteen hours) of work on the micro. I have! I also hope that you realize the need to know many different subjects in order to run computers. Good grades count too! So study hard, and I hope to see you again.

Best Wishes,

Doc

Teaching Music With The APPLE II Microcomputer

Dr. William Higgins
Messiah College
Grantham, Pennsylvania

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

COMPUTER-ASSISTED INSTRUCTION

in

MUSIC EDUCATION

for the

APPLE II MICROCOMPUTER

The Pennsylvania Music Educators In-Service Conference

Hershey, Pennsylvania

January 14, 1983

Presented by

Dr. William R. Higgins
Messiah College
Grantham, Pennsylvania

TOPICS

1. Micro Technology Unlimited "Song Pack"

2. Music Generators:

   A. Those which use the Apple II speaker.
      1. & Music (Nibble 2(8), 1981)
      2. Apple Fugue (Softside issue #34)
      3. Apple Music (Computer Components)
      4. Apple Music Maker (public domain)
      5. Appleodion (Rainbow Computing)
      6. Apple II Organ (Power Soft)
      7. Apple Piano (Creative Computing Software)
      8. Call -A.P.P.L.E. (Call-A.P.P.L.E. 5(6),1982)
      9. Electric Duet (Insoft)
      10. Forte (Softape)
      11. Keynote (Hayden Book Company)
      12. Maestro Music Generator (public domain)
      13. Music Box (Muse)
      14. Musicomp (Apple Computer Inc.)
      15. Music Machine (Softside 5(2), 1981)
      16. Music Maker (SubLogic)
      17. Music.9500 (Apple Programmer's Aid #1)
      18. Rainbow Music Generator (public domain)
      19. Songs in the Key of Apple (Hayden Book Co.)
      20. Soundware Music Box (United Soft. of America)
      21. Super Sound (Rainbow Computing)
      22. Tones (Creative Computing Software)

   B. Those which require additional hardware.

1.	Alf	(Alf Products)
2.	Analog Synthesizer Card	(Passport Designs)
3.	AppleDAC	(MTU)
4.	Music Board	(Programma)
5.	Music Composer	(Micro Music Inc.)
6.	Music Machine Nine	(Vista Media Products)
7.	Musicsystem	(Mountain Hardware)
8.	Super Music Synthesizer	(Applied Engineering)

3. Performance Systems

   1. Alpha Syntauri (Syntauri Corp.)
      A complete performance system including a keyboard, Mountain Hardware D/A boards and software.

   2. Compu-Music (RolandCorp US)
      A peripheral synthesizer with software which permits the user to write, change, store, and play compositions of up to eight voices.

3. Soundchaser (Passport Designs)
   A four octave polyphonic synthesizer keyboard, Mountain Hardware D/A boards, and software.

4. Music Printing Programs

   1. Manuscript (Notable Software)
   2. Notewriter (Passport Designs)

5. Micro Music Inc.

   A. General Music

      1. MUSICAL TERMS - Identification, spelling, and recall of commonly used general music terms (melody, harmony, instrumental forms and style, vocal forms, rhythm, notation, tone color, and dynamics).

      2. MUSIC SYMBOLS - Identification, spelling, and recall of commonly used music symbols.

      3. STANDARD INSTRUMENT NAMES - Identification, spelling, and recall of commonly used standard instrument names.

      4. FOREIGN INSTRUMENT NAMES - Identification, spelling, and recall of commonly used foreign instrument names.

      5. ITALIAN TERMS - Identification, spelling, and recall of commonly used Italian terms.

      6. COMPOSERS AND THEIR WORKS - Identification, spelling, and recall of composers and their significant works. Presented by historical period.

   B. Music Theory

      1. KEY SIGNATURE DRILLS - Includes exercises utilizing all the major and harmonic minor scales.

      2. DO-RE-MI - Aural identification of the individual degrees of a major scale.

      3. ARNOLD - Tone recognition and melodic memory skills. Enter the notes of an ever increasing melody.

4. NAME THAT TUNE - Identify the notes one by one until you can guess the tune.

5. PITCH IDENTIFICATION DRILLS WITHOUT ACCIDENTALS - Drills for the great, bass, and treble staffs. Also transposition for Bb, Eb, F and A instruments.

6. PITCH IDENTIFICATION DRILLS WITH ACCIDENTALS (as above)

7. MODE DRILLS - 12Ø scales from the major, minor and church modes are randomly presented for identification.

8. RHYTHM DRILLS - Whole notes to 16th notes (only dotted quarters and halves) randomly generated melodic rhythm patterns.

C. Extended Ear-Training

1. RHYTHMIC DICTATOR - Notating aurally presented rhythmic patterns.

2. MELODIOUS DICTATOR - Randomly generated melodic aural sequences are played back on the screen keyboard.

3. HARMONIOUS DICTATOR - Aural recognition of chord progressions using Roman numeral responses.

4. INTERVAL MANIA - Melodic or harmonic intervals in a "beat-the-clock" game format. Bass, treble, or full great staff.

5. CHORD MANIA - Aural/visual identification of 4-voice chords in any combination of qualities. Uses the "beat-the-clock" game technique.

6. HARMONY DRILLS - Aural recognition of diatonic chord progressions. Root position, 1st and 2nd inversions. Use with "The Harmony Drills" from William C. Brown publishers.

7. SIR WILLIAM WRONG-NOTE - Aural/visual identification of the wrong note in a four voice harmonic context. Major, minor, diminished, and augmented 6th chords.

D. Formal Music Theory

1. MUSIC IN THEORY AND PRACTICE - Eleven disks coordinate with and provide answers and instruction to the objective questions in the second edition of Dr. Benward's text.

6. Apple Music Theory (Apple Computer Inc.)

Originally developed by the Minnesota Educational Computing Consortium. Provided with excellent documentation.

A. AURAL INTERVALS - Melodic aural intervals. High to low or low to high. Uses major, minor, perfect, and augmented intervals. Seconds through octaves.

B. COUNTING - Fill in the missing note value (single note) to complete the measure. Meters used include 2/4, 3/4, 4/4, 5/4, 6/4, 3/8, 6/8, 9/8, and 12/8.

C. ENHARMONICS - Give the simplest enharmonic name for the pitch displayed.

D. FIND THE HALF - Aural identification of where the half-step is in a given scale pattern (3 to 5 notes).

E. KEY SIGNATURES - Visual identification of major and minor key signatures.

F. MISSING NOTE - A randomly generated 5 note sequence is visually displayed with one note missing. The sequence is played and the missing note is aurally identified. Sequences are diatonic.

G. NAME THE NOTE - Visual identification of notes in the treble and/or bass clef with or without ledger lines and sharps and flats.

H. NOTE TYPES - Visual identification of note and/or rest values and how many beats each would get in common meter. Note: this program uses decimal fractions for parts of a beat i.e. a half beat equals .5 and a quarter of a beat equals .25.

I. RHYTHM - Aural/visual identification. The computer plays three versions of the visual notation. The student selects the correct aural/visual match-up.

J. RHYTHM PLAY - Performance of visual notation. The student "taps" the notated rhythm using the "n" key. The computer evaluates the response.

K. SCALES - Aural identification of scales. Presents major, all three minors, dorian, phrygian, lydian, and mixolydian modes.

L. SEVENTHS - Aural identification of arpeggiated 7th chords (5 types).

M. TERMS - Multiple choice exam of 57 (primarily Italian) commonly used musical terms.

N. TRIADS - Aural identification of arpeggiated major, minor, augmented, and diminished triads.

O. VISUAL INTERVALS - Visual identification of two note intervals in all keys.

P. WHOLE-HALF - Aural identification of melodic whole-step or half-step intervals.

Q. WRONG NOTE - Aural/visual identification of the incorrect note in a given 5 note passage.

7. Notable Software

A. MUSICAL MATCH-UP - Flip the cards to match chords and chord symbols. Aural, visual, or aural/visual presentation. Ten difficulty levels.

B. NOTE TRESPASSING - The measure fills up with notes one at a time. The student can eliminate the notes by identifying them by letter name. If the measure fills up the student loses. If all the notes are removed, the student wins. Three levels of difficulty. Treble, bass, and various c clefs.

C. APPLE-NOME - Turns your computer into a practice metronome.

8. Edutek Corporation

A. MUSICAL STAFF DRILL - With a graphic display of a piano keyboard and grand staff, a player matches a piano key with a note of the staff by moving an electronic ball.

B. FIND THE KEY - beginning music students learn piano notes by matching piano keys with their letter names.

C. MAKE A TUNE - The player chooses notes and tempo and the program creates a tune using them.

9. Electronic Courseware Systems

A. ELEMENTS OF MUSIC - A set of three separate programs on pitch names, pitches on the keyboard, and key signatures.

10. Computer Applications Tomorrow

A. MUSICAL COMPUTER ONE AND TWO - Seven programs which include: an introduction to music symbols and language, note reading in the treble and bass clefs, telling time, note values and rests, piano keyboard, dynamic and tempo markings, and signs and symbols.

11. Advanced Operating Systems

A. MUSIC GAMES - Twelve different programs covering ear training, note recognition and writing, rhythm practice, and listening.

12. Mary Bee Communications

A. NOTES - A program to develop speed and accuracy in identification of notes by letter name (drill and game)

13. Howard W. Sams

A. MUSIC GAMES - Twelve aural visual games for the development of ear training, listening, note recognition, and rhythm & counting.

14. Passport Designs

A. MUSIC TUTOR - A CAI program for the Soundchaser that teaches general music theory and helps develop listening skills (Intervals, Matching & Tuning Chords, and Melodic Games).

15. Syntauri

A. MUSICMASTER - A modular software package for basic musicianship. includes topics from simple note recognition to identification of complex melodies. All programs contain a teacher management system for record keeping and scoring.

16. Instrumental Music Education

A. CLARINET BY COMPUTER - A complete program of beginning clarinet instruction. Designed for college music education majors.

B. MICRO-BRASS - Brass instrument pitch/note/fingering association. Student fingers notes presented by the computer which tells if correct or incorrect. Two modes are game or tutor.

17. Programs for Drill of Factual Materials

A. DEAN'S LIST - (XPS, Inc.) - Permits the teacher to prepare quizes which present one of two matched pairs of words or phrases (i.e. Eroica ---- Beethoven).

B. DRILL INSTRUCTOR - (XPS, Inc.) - Teacher types in questions in any format. The computer quizes students in random order returning missed questions to the pool until all are answered correctly.

C. THE GAME SHOW - (Computer-Advanced Ideas) - Teacher provides questions and answers. A quiz format in the form of the T.V. show "Password". Superb graphics.

D. THE SHELL GAMES - (Apple Computer Inc.) - A group of three quiz programs which permit the teacher to make quiz games which provide motivation and keep statistics on the student's progress.

E. E-Z LEARNER - (Silicon Valley Systems) - A menu-driven program creates, stores and reviews questions and answers on the subject of the teacher's input.

18. Other Uses of Microcomputers for Music Instruction

A. Computer Controlled Synthesizers
   1. Alphasyntauri
   2. Passport Designs

B. Instructional Games
   1. Crypto-Gen - (XPS, Inc.) - make musical cryptograms.
   2. Wordsearch - (I.A.C) - Make wordsearches of musical terms, composers, instruments, etc.

C. Test Generating

   1. Teacher Create Series (Educational Courseware)
      a. Completion Tests
      b. Matching Tests
      c. True/False Tests
      d. Multiple Choice Tests

   2. Test Writer (Persimmon Software)

D. Grading

   1. APPLE GRADE BOOK - (J & S Software)
   2. APPLE GRADEBOOK - (Creative Computing)
   3. ELECTRIC GRADE BOOK - (Charles Mann)
   4. GRADEKEEPER - (Ziggurat)
   5. INSTRUCTOR GRADEBOOK - (Serendipity Systems)
   6. TEACHER'S GRADEBOOK - (Dynacomp)

19. Administrative Uses for Microcomputers

A. Administrative Forms
B. Attendance Records
C. Awards Systems
D. Budget Analysis and Preparation
E. Charting Shows
F. Cost Analysis
G. Equipment Control
H. Fund Raising Administration
I. Instrument Inventory
J. Keeping Track of Feeder System
K. Music and Supplies Ordering
L. Music Filing System
M. Music Printing
N. Recordings File System
O. Rehearsal Planning
P. Schedules and scheduling
Q. Selection of Music
R. Uniform Control
S. Word Processing

MUSIC SYSTEMS FOR THE APPLE II

Name	Company	Approximate Cost
Alf	Alf Products Inc 1448 Estes Denver Colorado 80215	$265.00
AppleDAC	Micro Technology Unlimited 2806 Hillsborough Street P.O. Box 12106 Raleigh North Carolina 27605	$ 89.00
Music Composer	Micro Music Inc. Musitronic Box 441 555 Park Drive Owatonna Minnesota 55060	$350.00
MusicSystem	Mountain Hardware 300 Harvey West Blvd Santa Cruz California 95060	$545.00
Super Music Synthesizer	Applied Engineering P.O. Box 470301 Dallas Texas 75247	$159.00

Digital Music Synthesizers & Keyboards for the Apple II

Alphasyntauri Syntauri Ltd. 3506 Waverly Street Palo Alto California 94306	Five octave polyphonic keyboard and software to be combined with the Mountain Computer or Alf Synthesizer boards.
RolandCorp US 2401 Saybrook Avenue Los Angeles California 90040	Eight voice music synthesizer with seven voice electronic percussion. No keyboard. Music is programed from computer.
Passport Designs Box 478 La Honda California 94020	Polyphonic hybrid music system for the Apple II. Composer software and graphics. Control of envelope.

SELECTING THE RIGHT MICROCOMPUTER

Articles Which Compare Microcomputer Features

1. "Comparison of Major Popular Microcomputers". InfoWorld, 4(31), August 2, 1982.

2. "Home Computers: A Guide for Bewildered Buyers". Changing Times, 36(8), August 1982.

3. "Choosing the Best Computer for You". Money, 11(11), November 1982.

4. "Computer Specification Chart". School Product News, June 1982.

SELECTING THE MICROCOMPUTER

Recommended Apple System for Music Instruction & Administration

Note: Apple Computers, Inc. makes a "black" version of their microcomputer which is sold by Bell & Howell as the "Bell & Howell Microcomputer System". All Apple components listed below may be substituted with similiar Bell & Howell equipment.

It should be noted that all hardware and software discussed here will work with the FRANKLIN ACE 1000 microcomputer.

Recommended Minimum System

1.	Apple II Plus 48k Microcomputer	(ca. $1295)
2.	Apple Disk II with controller	(ca. $ 595)
3.	Microsoft 16K RAM Card	(ca. $ 169)
4.	Sanyo 9" B & W Monitor	(ca. $ 149)

Note: To use much of the quality music software or simply to have quality music reproduction, you will need to purchase one of the "Music Systems for the Apple II". Recommend: Micro Music's DAC Board (ca $175).

Valuable Extras

5.	A second disk drive	(ca. $495)
6.	External Speaker	(ca. $ 15)
7.	Dot matrix printer with graphics	(ca. $595)

Note: Remember that sufficient GOOD quality software will cost as much or more than the microcomputer.

SELECTED BIBLIOGRAPHY OF PERIODICALS

GENERAL PERIODICALS

Byte (Hardware Oriented)
P.O. Box 590
Martinsville
New Jersey 08836
($19/year)

Compute (broad coverage)
P.O. Box 5406
Greensboro
North Carolina 27403
($20/year)

Creative Computing (Software, Hardware, Uses, Reviews)
P.O. Box 789-M
Morristown
New Jersey 07960
($24.97/year)

Kilobaud Microcomputing (Software, Hardware, Uses)
P.O. Box 997
Farmingdale
New York 11737
($25/year)

Popular Computing
P.O Box 307
Martinsville
New Jersey 08836
($15/year)

Softside
100 Pine Street
Holmes
Pennsylvania 19043

APPLE ORIENTED PERIODICALS

Apple Educator's Newsletter (Educational Applications)
9525 Lucerne
Ventura
California 93003

Apple Orchard (Users Group Publication)
P.O. Box 2227
Seattle
Washington 98111
($10/year)

Call-A.P.P.L.E. (Users Group Publication)
304 Main Avenue South
Suite 300
Renton
Washington 980055
(Onetime $25 membership + $20/year Subscription)

inCider
Wayne Green, Inc.
P.O. Box 911
Farmingdale
New York 11737
($25/year)

Micro
P.O. Box 6502
Chelmsford
Massachusetts 01824
($18/year)

Nibble
P.O. Box 325
Lincoln
Massachusetts 01773
($20/year)

PERIODICALS FREE TO APPLE COMPUTER OWNERS

Apple Education News (What people are doing in education with Apple II microcomputers)
10260 Bandley Drive
Cupertino
California 95014

Softalk (Articles and Reviews)
10432 Burbank Boulevard
North Hollywood
California 91601

PERIODICALS WHICH FOCUS ON COMPUTERS IN EDUCATION

Apple Educator's Newsletter
9525 Lucerne Street
Ventura
California 93004
($15/year)

Classroom Computer News
341 Mt. Auburn Street
Watertown
Massachusetts 02172

The Computing Teacher
ICCE Department 4A
Department of Computer & Information Service
University of Oregon
Eugene
Oregon 97403

Educational Technology
140 Sylvan Avenue
Englewood Cliffs
New Jersey 07632

PERIODICALS WHICH FOCUS ON COMPUTERS IN MUSIC

Computer Music Journal
Massachusetts Institute of Tehnology Press
28 Carleton Street
Cambridge
Massachusetts 02142

Small Computers in the Arts News
Small Systems Press
404 S. Quince Street
Philadelphia
Pennsylvania 19147

PERIODICALS WHICH REVIEW SOFTWARE

Journal of Courseware Review
The Apple Education Foundation
20525 Mariani Avenue
Cupertino
California 95014

Peelings II
P.O. Box 188
Las Cruces
New Mexico 88044
($15/year)

Softalk (Fasttalk Section)
10432 Burbank Boulevard
North Hollywood
California 91601
($18/year)

APPLE II SOFTWARE REFERENCES

Stanton, J., Wells, R.P., & Rochowansky, S. THE ADDISON-WESLEY BOOK OF APPLE COMPUTER SOFTWARE 1983. Los Angles: The Book Company, 1983.

Vanloves, G. & Love,R. 1983 APPLE II/III SOFTWARE DIRECTORY. Overland Park, Ks.: Vital Information Inc., 1983.

---------. EDUCATIONAL SOFTWARE DIRECTORY APPLE II EDITION. Manchaca, Tx.: Sterling Publishing Company, 1982.

---------. APPLE SOFTWARE DIRECTORY/EDUCATION. Chicago: WIDL Video, 1982.

---------. 1983 CLASSROOM COMPUTER-NEWS DIRECTORY OF EDUCATIONAL COMPUTING RESOURCES. Watertown, Mass.: International Educations, Inc., 1982.

COMPLETE APPLE II BOOK LIST

Robert Broedel
P.O. Box 20049
Tallahassee
Florida 32304

(Nibble 3(1), 1982)

SELECTED BIBLIOGRAPHY OF

Specific Periodical Issues Devoted to Educational Applications

1. "Trends in Computer-assisted Instruction". Educational Technology, 18(4), April 1978.

2. "Educational Computing". Creative Computing, 5(9), September 1979.

3. "Microcomputers in Education". Educational Technology, 19(10), October 1979.

4. "Microcomputers in Education". Kilobaud Microcomputing, #42 June 1980.

5. "Computers and Education". Byte, 5(7), July 1980.

6. "Emphasis on Education". Creative Computing, 7(3), March 1981.

7. "Learning with Computers". Kilobaud Microcomputing, #57, Kilobaud Microcomputing, September 1981.

8. "Educational Applications". Creative Computing, 7(10), October 1981.

9. "Computers in the Schools". Educational Technology, 21(10), October 1981.

10. "Computers in Education". Apple Orchard, 3(1), March-April 1982.

11. "Computers in Education". Creative Computing, 8(4), April 1982.

12. "Microcomputers in the Classroom". Today's Education, 71(2) April-May 1982.

13. "Resource Sharing". T.H.E. Journal, 9(4), May 1982.

14. "A Practical Guide to Computers in Education". Creative Computing, 8(10), October 1982.

SELECTED BIBLIOGRAPHY OF

Specific Periodical Issues Devoted to Music Applications

1. "Scroll Over, Beethoven: Tune Up Your Micro for Music". Kilobaud Microcomputing, #47, December 1980.

2. "Computer Music". Byte, 5(4), April 1980.

3. "Special Computer Graphics and Music Issue". Creative Computing, 6(6), June 1980.

4. "In-Depth Evaluation of Nine Music Editors". Creative Computing, 7(2), February 1981.

5. "Music in the Micro". Softside, 5(2), November 1981.

6. "Pitch and Rhythm on the Apple". Call-A.P.P.L.E. 5(6), June 1982.

7. "Computers in the Arts and Sciences". Byte, 7(7), July 1982.

8. "The Sound of the Future". Softside, 6(1), #34, September 1982.

9. "Music, Sound and the Personal Computer". Compute, 5(1), January 1983.

10. "Technology". Music Educators Journal, 69(5), January 1983.

ORGANIZATIONS PROMOTING COMPUTERS AND MUSIC

Computer Music Association
911 22nd Avenue South #181
Minneapolis
Minnesota 55404

National Consortium for Computer-Based Music Instruction
(NCCBMI)
Dr. G David Peters, President
Music Education Division
School of Music
University of Illinois
Urbana
Illinois 61801

Small Computers in the Arts
P.O. Box 1954
Philadelphia
Pennsylvania 19105

ORGANIZATIONS PROMOTING COMPUTERS IN GENERAL EDUCATION INCLUDING MUSIC

International Council for Computers in Education (ICCE)
Department of Computer & Information Science
University of Oregon
Eugene
Oregon 97403

Minnesota Educational Computing Consortium
2520 Broadway Drive
Lauderdale
Minnesota 55113

Northwest Regional Educational Laboratory (NREL)
300 Southwest 6th Avenue
Portland
Oregon 97204

GENERAL EDUCATION AND ADMINISTRATION SOFTWARE PUBLISHERS

PROGRAM	COMPANY
THE DEAN'S LIST THE DRILL INSTRUCTOR (Drill programs)	XPS, Inc. 323 York Road Carlisle Pennsylvania 17013
THE GAME SHOW (Drill Game)	Computer Advanced Ideas, Inc. 1442A Walnut Street Suite 341 Berkeley California 94709
E-Z LEARNER (Drill Programs)	Silicon Valley Systems 1625 El Camino Real #4 Belmont California 94002
TEACHER CREATE SERIES (Test Maker Programs)	Educational Courseware 3 Nappa Lane Westport Connecticut 06880
APPLE GRADEBOOK (Electronic Grading)	Creative Computing 39 East Hanover Avenue Morris Plains New Jersey 07950
APPLE GRADE BOOK (Electronic Grading)	J & S Software 140 Reid Avenue Port Washington New York 11050
ELECTRIC GRADEBOOK (Electronic Grading)	Charles Mann 7584 San Remo Trail Yucca Valley California 92284
GRADEKEEPER (Electronic Grading)	Ziggurat Box 453 Arlington Heights Illinois 60006

(General software continued)

TEACHER'S GRADEBOOK (Electronic Grading)	Dynacomp 1427 Monroe Avenue Rochester New York 14618
TEST WRITER (Creates Exams)	Persimmon Software 502 Savannah Street Greensboro North Carolina 27406
WORDSEARCH (Public Domain Programs)	Apple Avocation Alliance (AAA) 721 Pike Street Cheyenne Wyoming 82009
PFS: PERSONAL FILING SYSTEM (Easy to use Record Keeping)	Software Publishing 2021 Landings Drive Mountain View California 94043
DB MASTER (Best Data Base System)	Stoneware Microcomputer Products 50 Belvedere Street San Rafael California 94043
VISICALC 3.3 (Budgets, Fund Raising, Forecasting)	Personal Software 1330 Bordeaux Drive Sunnyvale California 94901
SCREENWRITER II (Word Processor)	On-Line Systems 36575 Mudge Ranch Road Coarsegold California 93614
MAGIC WINDOW (Easy to Use Word Processor)	Artsci, Inc. 10432 Burbank Boulevard North Hollywood California 91601

MUSIC EDUCATION SOFTWARE PUBLISHERS

Advanced Operating Systems
4300 West 62nd Street
P.O. Box 7092
Indianapolis
Indiana 46206

Alf Products
1448 Estes
Denver
Colorado 80215

Apple Computer Inc.
10260 Bandley Drive
Cupertino
California 95014

Creative Computing
Software Department
39 E. Hanover Avenue
Morris Plains
New Jersey 07950

Computer Applications Tomorrow
P.O. Box 605
Birmingham
Michigan 48012

Dandelion Micro Products
P.O. Box 615
Mount Vernon
Illinois 62864

Edutek
415 Cambridge #14
Palo Alto
California 94306

Electronic Courseware Systems
P.O. Box 2374, Station A
Champaign
Illinois 61820

Hayden Book Company
50 Essex Street
Rochelle Park
New Jersey 07662

Insoft
10175 Southwest Barbur Blvd
Suite 202B
Portland
Oregon 97219

Micro-Music, Inc.
Division of Musitronic
555 Park Drive
P.O. Box 441
Owatonna
Minnesota 55060

Micro Technology Unlimited
2806 Hillsborough Street
P.O. Box 12106
Raleigh
North Carolina 27605

Minnesota Educational Computing Consortium (MECC)
2520 Broadway Drive
Lauderdale
Minnesota 55113

Muse Software
330 North Charles Street
Baltimore
Maryland 21201

Notable Software
P.O. Box 1556
Philadelphia
Pennsylvania 19105

Nibble
P.O. Box 325
Lincoln
Massachusetts 01773

Opportunities for Learning, Inc.
8950 Lurline Avenue
Chatsworth
California 91311

Passport Designs
116 North Cabrillo Highway
Half Moon Bay
California 94019

Powersoft
Box 157
Pitman
New Jersey 08071

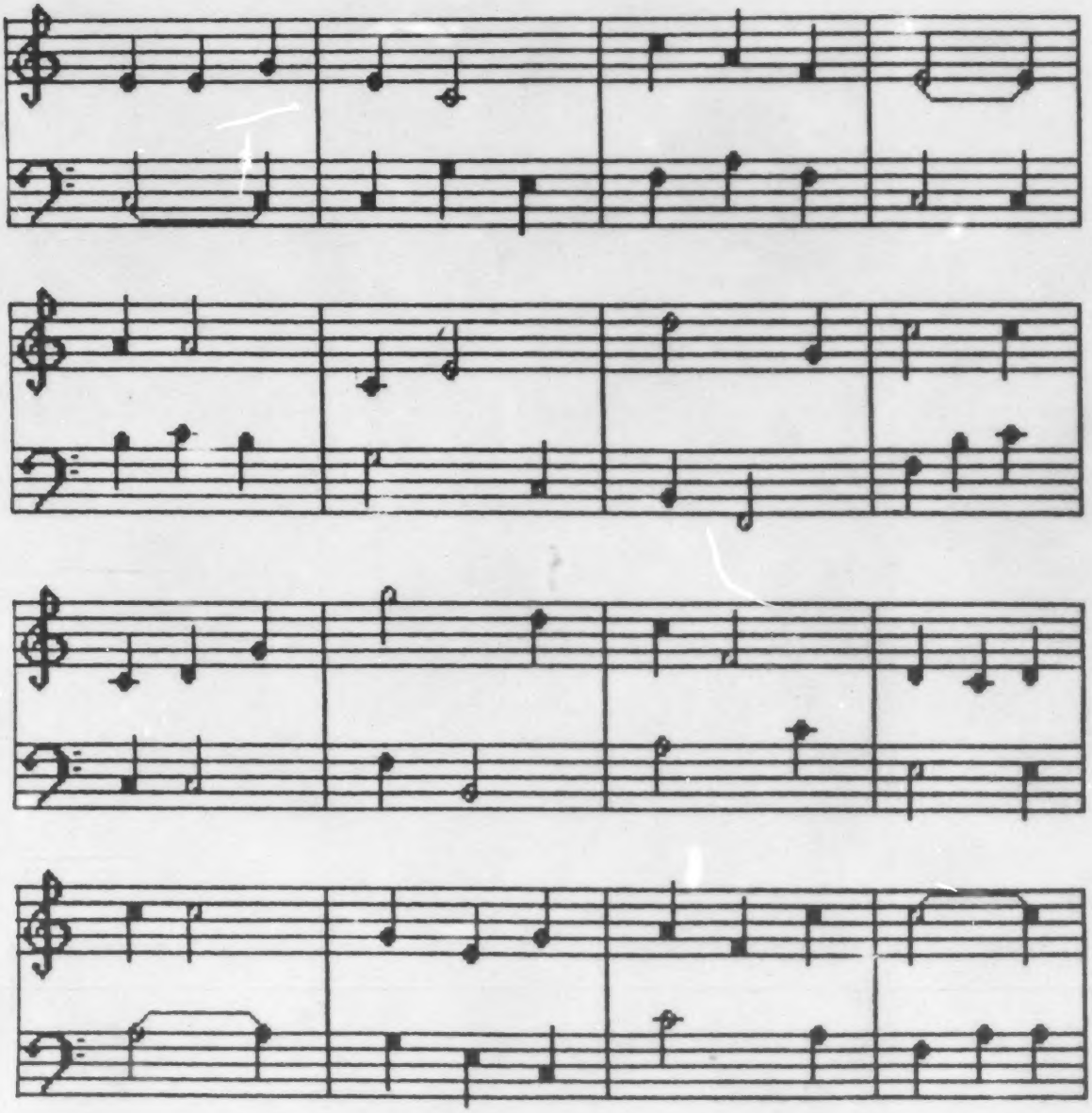
Rainbow Computing
19517 Business Center Drive
Northridge
California 91324

Serendipity Systems
419 West Seneca Street
Ithaca
New York 14850

Softside
6 South Street
Milford
New Hampshire 03055

SubLogic
713 Edgebrook Drive
Champaign
Illinois 61820

Systems Design Laboratory
2612 Artesia Blvd.
Suite B
Redondo Beach
California 90278

solbuqjes

Note: This Crab Canon was composed by Michael Keith's Apple II program NCC (Numerical Canon Composition. The printing was done with an Epson MX-80 printer using Rebecca Mercuri's program MANUSCRIPT.

The Design of a Microcomputer Data Base for Educators

Ms. Jean Howard
Mr. Ronald Aitken
Mr. Timothy Robinson
Instructional Systems Graduate Students
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

THE DESIGN OF A MICROCOMPUTER DATA BASE FOR EDUCATORS

presented at

The Second Annual Microcomputer Conference
The College of Education & Department of Computer Science
The Pennsylvania State University

March 11-12, 1983

Presented by

Carol Jean Howard
Ronald Aitken
Timothy Robinson

THE DESIGN OF A MICROCOMPUTER DATA BASE FOR EDUCATORS

Introduction

Looking at the global picture, Computer Managed Instruction (CMI) is the instructional support functions conducted by the computer. (PDE) These support functions are capable of testing, diagnoses, prescribing, assigning, facilitating study, reporting, evaluating, record keeping, schedule monitoring, time, resource and space management, etc. The "core element" of all CMI systems is the computer component of the data base. (Baker, 1978) The data base is the storage facility for all the information in the computer. The data base contains tne information needed for the support functions. In essence, the data base in the computer determines what types of information one can access. This necessitates careful and thoughtful planning during the design phase.

The design phase consists of two parts: logical and physical. The logical data base is "all of te data needed to correctly produce the output(s) desired." (Higgins,1979) The physical data base is concerned with the implementation of the logical data base part (e.g., arrangement of data, limitations on the system, etc.).

Purpose

The intent of this paper will address the designing of a logical data base along with insights for the physical construction of two data base systems: Student Data Base for the Instructional Systems Division of the College of Education and an Inventory Data Base ~~of students~~ for a Alumni Microcomputer Software Library.

Background

The model used for this paper is derived from the one described in David A. Higgins' book Program Design and Construction. The model is founded on the systematic approach using a top-down or "decomposition" process. (Higgins, 1979) The top-down process starts with the highest level of conceptualization proceeding incrementally in a step-by-step process down to the operatant level. The Warnier-Orr (W-O) diagram becomes the tool for this process. The W-O diagram is also known as "Output Oriented Analysis." (Higgins, 1979) This state-of-the art program design method is founded on the mathematical concept of set theories. The idea is to take an ordered collection of like objects (e.g., sets) and an ordered collection of like objects that are globally contained within some other set (e.g., subset) to produce set notation diagrams. These diagrams start with a goal and break down into specific "whats." (Higgins, 1979) The idea of set notation is a reverse style of program design (i.e., starting with and using the final or end product to determine the enabling elements that will be required to reach this end product). Set notion diagrams have the following functions:

1. the ability to show the process of sequence
2. the concept of hierarchy
3. process of repetation
4. alternation process structure (Higgins, 1979)

Examples of Set Notion

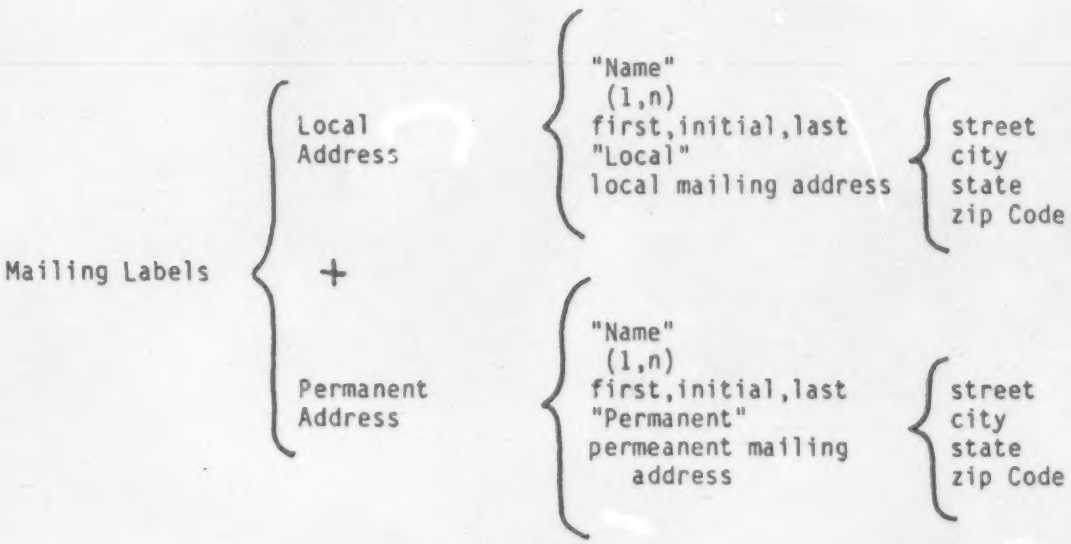


Figure 1.1 "Conditional... Exclusive OR" Set Notation

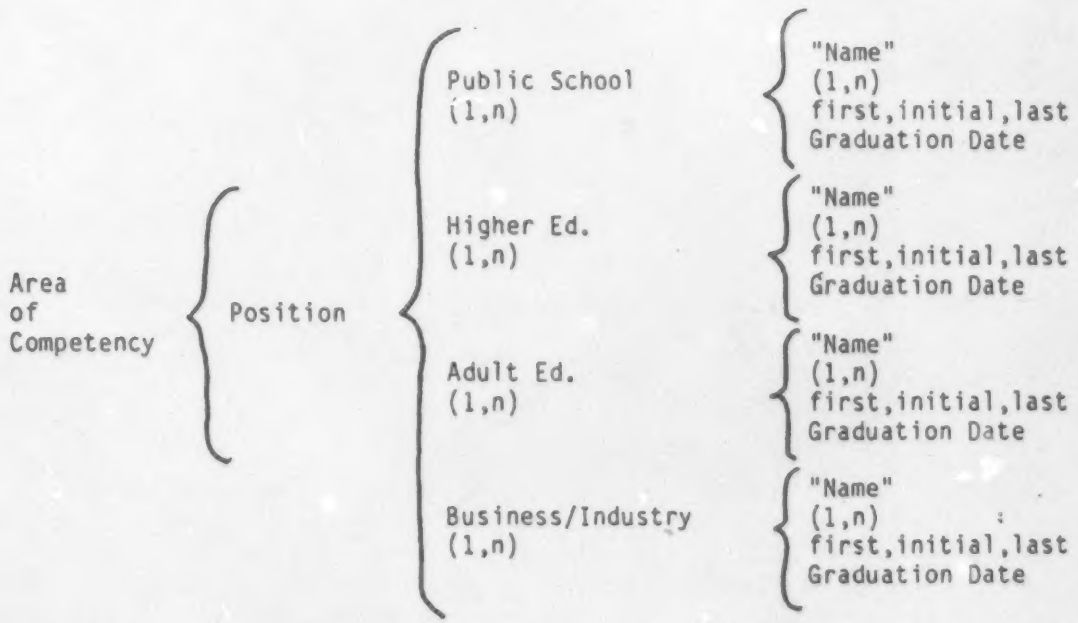


Figure 1.2 "Inclusive OR" Set Notation

Design: The Logical Data Base

STEP ONE - Define the Process

This step requires that the designer determine what is wanted as a final product or output. Thus, the designer is concerned with the question: "What do we want as a final product or output?" (Higgins, 1979)

Higgins suggests answering this question by defining a sample report for each output. This is an extremely important step since the data must be in the computer in order to produce an output, otherwise, any missing data will produce an incompete output. Information that is needed to complete an output is called a data structure. The data structure for Figure 1.1 are the individual's first name, middle initial, and last name, and street, city, state, zip code, (local and permanent). These data structures provide information for the output form, mailing labels. Higgins provides the following diagram of the output process.

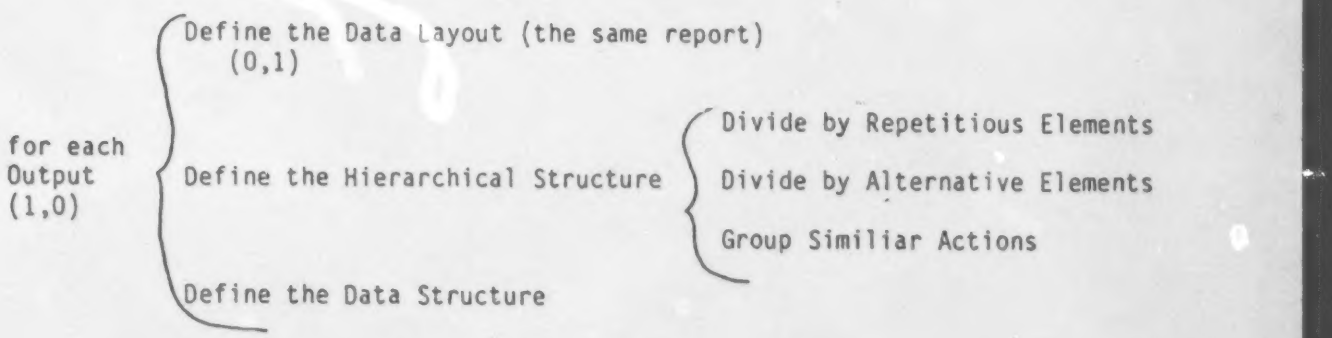

for each Output (1,0)
Define the Data Layout (the same report) (0,1)
Define the Hierarchical Structure
Divide by Repetitious Elements
Divide by Alternative Elements
Group Similiar Actions
Define the Data Structure

Figure 2.1 Definition of Output(s)

STEP TWO - Define the Logical Data Base (LDB)

A logical data base is all of the data that is needed to correctly produce the output(s) desired. (Higgins, 1979) Fundamentally, this step contains three parts: testing, mapping, and re-examining. The first part requires that all

data elements on a report are listed as data fields (actual numbers or representations in small letters) and titles and headings (listed in capital letters and enclosed in quotes)--See Figure 1.1.

The second part requires mapping the elements into the data structure. The question the designer is concerned with is "Where does this element occur in the data structure?" (Higgins, 1979) This part charts the relationship of all the data elements of the report with its logical data structure. An example can be seen in Figure 1.1 or Figure 1.2. The final part of this step requires a re-examination of the data elements. The designer must make the following checks:

1. Are all the data elements accounted for or is there a hidden hierarchy?
2. Are all the data elements necessary?
3. Do the data elements map into the data structure properly?
4. After all the above checks are completed, does the LDB still describe or produce the correct output? (Higgins,1979)

Once the above process is satisified (i.e., the LDB describes the correct output), the designer can move on to the second stage of design, the physical data base.

Design: Physical Data Base

This phase must follow the LDB for two reasons. First, the LDB must be defined before physical limitations can be encountered (i.e., logical design and system interface, timing, storage, etc.), and second, maintenance problems may surface due to system modifications. (i.e., each step upward in hardware modification will necessitate a change in software).

STEP THREE - Define Events

This step requires an examintion of all external environmental events that can effect the data elements. To examine these events, the designer needs to examine entities. Entities show how different parts of the LDB relate to other parts. Generally, entities are the names for the hierarchical brackets. The designer must be concerned with how the entity can began, identify events, and discover how the entity can end. The designer identifies events by finding the affecting atrributes and listing how the attributes can change. The advantage of event analysis is two-fold. First, can be discovered additional requirements for the output that were previously unknown, and second, it provides the opportunity to create facilities that can change some attributes once they are in the data.

STEP FOUR - Develop the Physical Data Base

The physcial data base is the creation of the LDB into a physical form. At this stage the arrangement of data becomes a reality. Data are arranged in the form of files and records. A file consists of a collection of an individuals records containing the same kind of information. A record is information unique to the individual. The designer uses the defined input to assemble information into a useable form. This builds "user friendliness" into the files.

STEP FIVE - Design the Logical Process

This step involves the concept of mapping. The idea of mapping is to look at the LDB and describe what action must take place to produce the required outputs. Each hierarchical segment is broken down into a beginning, middle, and end. The actions that are defined in mapping identify the information to be

built into the design from the end-highest-level to the beginning-highest-level. This will complete the design process of the physical data base by integrating the details of the process.

STEP SIX - Design the Physical Process

There are two approaches for physcial data base design. The designer has the choice of creating the program from scratch through the use of a computer program langauge (e.g. BASIC, PASCAL, etc.) or purchasing a pre-packaged software program. There are various advantages and disadvantages to both alternatives. The designer's decision is determined by many internal and external environmental contraints.

Summary

In retrospect, the data base design being the "core element" of any Computer Managed Instructional system will determine, by a large measure, the success of the system and its "user friendly" accessability to the information. Therefore, we hope that this paper has made you aware of the importance of the design of any computer data base and, at the same time, demonstrated that it is not difficult to design one if one goes about it in a logical and systematic way.

REFERENCES

Baker, Frank B., Computer Managed Instruction: Theory and Practice, Englewood Cliffs: Prentice-Hall, Inc., 1978.

Higgins, David A., Program Design and Construction, Englewood Cliffs: Prentice-Hall, Inc., 1979.

A Computer Programming Curriculum for High School (I, II, III)

Mr. Newton Kulp
Central Bucks High School-West
Doylestown, Pennsylvania

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

COURSE MATERIALS

FOR

COMPUTER PROGRAMING I

COURSE

AS TAUGHT AT

CENTRAL BUCKS HIGH SCHOOL - WEST

DOYLESTOWN, PA.

Written by

Newton E. Kulp

Answer the following questions about the programm

1.) Did the computer do anything with line 100 on the screen ?

2.) What do you think lines 110,130,160, and 190 are doing?

3.) What are lines 180 and 210 allowing you to do?

Name _____

Exercise 1

You type:

```
    NEW
10  Let x+1
10  LET X = 1
20  PRINT X
30  LET X=X+1
40  GOTO 20
RUN
```

Computer types: (don't forget to press the BREAK button to stop the computer)

Describe outcome:

Exercise 2

You type:

```
20 PRINT X,
RUN
```

Computer types: (don't forget to press BREAK)

Describe outcome:

Exercise 3

You type:

```
20 PRINT X;
RUN
```

Computer types:

Describe outcome:

Exercise 4

Type: LIST

Computer outcome was (describe - do not copy)

Which line 20 ddd the computer print?

How do you think you change a line to something else?

What do you think line 30 does?

What do you think line 40 does?

Name _____

Exercise 1

You type:
```
NEW
10  PRINT "LOVE"
20  PRINT "WAR"
15  PRINT "AND"
RUN
```

Computer types:

Describe what you think the computer did.

Exercise 2

You type:
```
NEW
10 PRINT "LOVE";
20 PRINT "AND WAR"
RUN
```

Computer types:

Describe the outcome:

Exercise 3

You type:
```
NEW
10 PRINT "LOVE ";
15 PRINT "AND WAR"
RUN
```

Computer types:

Describe the outcome:

Exercise 4

You type:
```
NEW
100 PRINT  "5"
150 PRINT  "10"
200 PRINT  "15"
250 PRINT  "20"
```

The computer types:

Describe the outcome

Exercise 5

You type:

```
NEW
20 PRINT "5";
21 PRINT "10";
22 PRINT "15";
23 PRINT "20";
RUN
```

The computer prints on the screen:

Describe the outcome:

Exercise 6

You Type:

```
NEW
55 PRINT"5",
60 PRINT "10",
70 PRINT "15",
80 PRINT "20",
90 PRINT "25",
RUN
```

The computer prints on ths screen:

Describe the outcome:

Computer Programming I
Literacy Introduction
Page 1

Name _____

EXERCISE 1

You type: PRINT 20 + 65 (Enter)

Computer responds with: _____

Describe what you think the computer did.

EXERCISE 2

You type: PRINT 10 - 15 (Enter)

Computer responds with: _____

Describe what you think the computer did.

EXERCISE 3

You type: PRINT 5 * 15 (Enter)

Computer responds with: _____

Describe what you think the computer did.

EXERCISE 4

You type : PRINT 9/5 (Enter)

Computer respond s with: _____

Describe what you think the computer did.

EXERCISE 5

You type : (Make up your own problem using at least 3 of the above symbols) Put your problem in the following blank

_____

You think the answer should be: _____

Computer responds with(after pressing enter): _____

BEST COPY AVAILABLE

Exercise 1

You type: PRINT "20 + 45"

Computer responds with: _____

Describe what you think the computer did.

Exercise 2

You type: PRINT 3↑3

Computer responds with: _____

Describe what you think the computer did.

Exercise 3

You type: PRINT "_____" (put your name where the line is placed)

Computer types: _____

Describe what you think the computer did.

Exercise 4

You type: PRINT "I LOVE MY COMPUTER."

Computer types: _____

Describe the outcome.

Exercise 5

You type: try and write a line that would display on the screen the following:

I WAS ABLE TO MAKE THE COMPUTER PRINT THIS LINE

Write your guess on this line before you type it into the computer.

_____

Did the computer do what you want it to on the first try? (yes or no) ____

BEST COPY AVAILABLE

COMPUTER PROGRAMING I

COURSE OUTLINE

Week	Concept
1-2	Overview of the course and exercises in keyboard familiarity via simple games and programs Overview of the progamming aspects vai literacy sheets
3	Formalization of concpts presented on the literacy sheets Presentation of variables, operations, input, output, and line numbering. Project 2 - Write a program to input two numbers, then add, subtract, multiply, and divide them and print the results. Save project on tape or disk Retrieve project and add headings (literal prints) Add simple print controls (ie. , and ;) Add unconditional branch to do more than one set of values (GOTO)
4	Introduction to the conditional branching statement (IF-THEN) In class project done by the instructor to find the sum of the first 50 natural numbers. Project 3 - modify the above in class progam to find the sum of the first 50 odd/even natural numbers After class finished this project rewrite the program to input a beginning number, an ending numbers, and and increment, then find the sum (arithmetic series program)
5	While working on Project 3 editing techniques and error message analysis is presented
6	Looping and repetitive processes introducted (FOR-NEXT) In class use the two examples from IF-THEN and convert to FOR-NEXT Project 4 - see handout
7	Introduction to preprogammed functions In class write a simple program to input a number and then print out its absolute value function without using the ABS function Present functions already in the computer and use examples of each in a program (ABS,INT,SQR,SIN,SOC,TAN,ATN) Project 5 - see handout (these are harder projects thus given about 7 school days to finish)
8	Review and midterm examination
9	Introduction to subscripted variables (single dimension arrays) DIM and subscript Use books on a shelf concept to illustrate how an array works Write a program in class to input numbers, print them across the screen, then ask which number by location wanted to print and then print it and its location number Write a program in class to find graph points of a polynominal function by inputing the coefficients via an array (see sheet) Introduction to READ-DATA statements Project 6 - see handout

WEEK	CONCEPT
10	Printing Controls and how to use them to enhance output TAB and further explanation of , and ; with exact column locations (see sheet) PRINT USING (see sheet) STRING$(,) Project 7 - Plumbing Sales project - see sheet
11-12	Introduction to two dimensional arrays (matrices) Expand DIM and subscripts Use library bookshelf idea to expand a visual representation of a matrice as well as cartesian coordinate plance (rows and columns) Assign first two parts of project 8 - Questionnaire Analysis Explain how to calculate weighted averages Write in class an interpolation program Write in class a traffic survey program Assign rest of questionnaire analysis project
13-14	Miscellaneous progrmming techniques Random number generator (RND) Multiple branch staetement (ON - GOTO) Subroutines (use as example the interpolation program written previously and modify to be subroutine) GOSUB - RETURN Write in class the card dealing program which combines matrices, RND, ON- GOTO, printing controls, and other techniques presented in course Project 9 - free project - the student may do any project they wish after getting approval from the instructor
15	Itteration processes (optional) Use guess my number game to illustrate concept and write this program in class Explain hypothesis testing and error limits Project 10 - see sheet
16	Final project - instructor assigns from sheet or from other sources a project the student must do without help from instructor. Studen may also suggest a project and do it instead if instructor approves
17	Review for final and final exam

Name _____

Exercise 1

You type the following (this is called a program)

```
 NEW
100 REM CHECKBOOK BALANCING PROGRAM
110 PRINT
120 PRINT "I WILL BALANCE YOUR CHECKBOOK"
130 PRINT
140 PRINT " ENTER CHECKS AS NEGATIVE NUMBERS"
150 PRINT " ENTER DEPOSITS AS POSITIVE NUMBERS"
160 PRINT
170 PRINT "OLD BALANCE";
180 INPUT B
190 PRINT
200 PRINT "CHECK OR DEPOSIT AMOUNT ";
210 INPUT A
220 LET B=B+A
230 PRINT "NEW BALANCE;"B
240 GOTO 190
RUN
```

Computer types:

(enter a number which would be a possible answer to the question)

Place your response here; _____

Computer then types:

Enter a number which would be a possible answer to this question.
Do this for five (5) times and record outcome exactly as shown on screeen below.

Remember to press the BREAK button when you are finished to release the computer from the program control.

Did the computer print the correct statement?

You type:

RUN

What did the computer do this time?

Complete the program like you did before.

What do you think happened when you typed RUN the second time in this exercise?

What do you think you must do if you want the computer to repeat something it has just completed?

BEST COPY AVAILABLE

Name _____

Exercise 1

You Type:

```
NEW
10 REM CHANGE CELSIUS TO FAHRENHEIR
20 PRINT  "YOU ENTER DEGREES CELSIUS"
30 PRINT  "I WILL GIVE DEGREES FAHRENHEIT"
40 PRINT
50 PRINT "DEGREES CELSIUS";
60 INPUT C
70 PRINT "DEGREES FAHRENHEIT =";
80 PRINT (9/5)*C + 32
90 GOTO 40
RUN
```

Computer types:

Describe outcome:

Exercise 2

You type:

```
NEW
110 PRINT "ENTER A NUMBER AND I WILL TELL YOU IF IT IS"
120 PRINT "LESS THAN, EQUAL TO, OR GREATER THAN 100"
130 PRINT
140 PRINT "WHAT IS YOUR NUMBER";
150 INPUT N
160 IF N<100 THEN PRINT "YOUR NUMBER IS LESS THAN 100."
170 IF N = 100 THEN PRINT "YOUR NUMBER IS EQUAL TO 100."
180 PRINT "YOUR NUMBER IS GREATER THAN 100."
RUN
```

Computer types:

Then you type in any number - put your number in the blank _____

Computer types:

BEST COPY AVAILABLE

Did the computer print the correct statement?

You type:

RUN

What did the computer do this time?

Complete the program like you did before.

What do you think happened when you typed RUN the second time in this exercise?

What do you think you must do if you want the computer to repeat something it has just completed?

BEST COPY AVAILABLE

Name _____

Exercise 1

Something new for you to do first. Study the following program carefully before you go to the computer. Answer the questions below it and bring to the instructor first. Only after he has seen your answers will you be allowed to continue.

```
110 PRINT "YOU ENTER A NUMBER AND I WILL TELL YOU WHETHER IT IS"
120 PRINT "POSITIVE, NEGATIVE, OR ZERO."
130  PRINT
140  PRINT "WHAT IS YOUR NUMBER";
150 INPUT N
160 IF N < 0 THEN PRINT "YOUR NUMBER IS NEGATIVE."
170 IF N = 0 THEN PRINT "YOU TYPED IN ZERO."
180 IF N > 0 THEN PRINT "YOU TYPED A NUMBER THAT IS POSITIVE."
```

Explain in a simple sentence what you think this program will do.

What do you think line 150 allows you to do?

What do you think line 160 is doing?

If you would type the number 30 when the computer asks you for a number, what line number do you think the computer would print on the screen?

_____

STOP - SHOW YOUR ANSWERS TO THE INSTRUCTOR BEFORE YOU CONTINUE.

Instructors initials proving seen sheet. _____

Now go to a computer terminal and type the above program in and repeat 4 times, making sure you type in a postive, negative, and zero number for one of the times.

Remember to type NEW before you type in the program.
Remeber to type RUN when finish typing and to restart the program.

Name _____

Exercise 1

You Type:

```
NEW
5  REM  (Type your name here)
10 X = 1
20 FOR C = 1 TO X
30 PRINT "*";
40 NEXT C
50 PRINT
60 X = X + 1
70 IF X > 10 THEN END
80 GOTO 20
RUN
```

Computer types:

Describe what happens:

Exercise 2

You type:

```
NEW
5 PRINT"YOU CAN DRAW A RECTANGLE THAT IS A MAXIMUM OF 63 UNITS LONG
   AND 15 UNITS WIDE."
10 PRINT"HOW WIDE A RECTANGLE DO YOU WISH TO DRAW";
20 INPUT W
30 PRINT"HOW LONG A RECTANGLE DO YOU WISH TO DRAW";
40 INPUT L
50 CLS
60 FOR J=1 TO L:PRINT"*";:NEXT J:PRINT
70 FOR K=1 TO W:PRINT"*";TAB(L);"*":NEXT K
80 FOR M=1 TO L:PRINT"*";:NEXT M:PRINT
90 END
```

Computer prints on the screen:

BEST COPY AVAILABLE

make the figure on the screen look like a square. (Hint - they are not equal since the spacing per character is different across and down.)

From what you have learned from these sheets see if you can write a simple program that would draw a picture or design on the screen. A word of warning here, keep it simple or you may find your self in trouble (KISS).

BEST COPY AVAILABLE

Edit Mode Subcommands and Functions

Sub-Command	Function	Page
(ENTER)	End editing and return to Command Mode.	196
(SHIFT) (↑)	Escape from X, L, and H subcommands and remain in Edit Mode.	197
n(SPACEBAR)	Move cursor *n* spaces to right.	196
n(←)	Move cursor *n* spaces to left.	197
(L)	List remainder of program line and return to beginning of line.	197
(X)	List remainder of program line, move cursor to end of line, and start Insert subcommand.	198
(I)	Insert the following sequence of characters at current cursor position; use Escape to exit this subcommand.	198
(A)	Cancel changes and return cursor to beginning of line	199
(E)	End editing, save all changes and return to Command Mode.	199
(Q)	End editing, cancel all changes made and return to Command Mode.	199
(H)	Delete remainder of line and insert following sequence of characters; use Escape to exit this subcommand.	199
n(D)	Delete specified number of characters *n* beginning at current cursor position.	200
n(C)	Change (or replace) the specified number of characters *n* using the next *n* characters entered.	200
n(S)*c*	Move cursor to *n*th occurrence of character *c*, counting from current cursor position.	201
n(K)*c*	Delete all characters from current cursor position up to *n*th occurrence of character *c*, counting from current cursor position.	202

PRINT USING *string; item list*

This statement allows you to specify a format for printing string and numeric values. It can be used in many applications such as printing report headings, accounting reports, checks, or wherever a specific print format is required.

The PRINT USING statement uses the following format:

PRINT USING *string ; value*

String and *value* may be expressed as variables or constants. This statement will print the expression contained in the string, inserting the numeric value shown to the right of the semicolon as specified by the field specifiers.

The following field specifiers may be used in the string:

#	This sign specifies the position of each digit located in the numeric value. The number of # signs you use establishes the numeric field. If the numeric field is greater than the number of digits in the numeric value, then the unused field positions to the left of the number will be displayed as spaces and those to the right of the decimal point will be displayed as zeros.
.	The decimal point can be placed anywhere in the numeric field established by the # sign. Rounding-off will take place when digits to the right of the decimal point are suppressed.
,	The comma — when placed in any position between the first digit and the decimal point — will display a comma to the left of every third digit as required. The comma establishes an additional position in the field.
**	Two asterisks placed at the beginning of the field will cause all unused positions to the left of the decimal to be filled with asterisks. The two asterisks will establish two more positions in the field.
$	A dollar-sign will be printed ahead of the number.
$$	Two dollar signs placed at the beginning of the field will act as a floating dollar sign. That is, it will occupy the first position preceding the number.
**$	If these three signs are used at the beginning of the field, then the vacant positions to the left of the number will be filled by the * sign and the $ sign will again position itself in the first position preceding the number.
↑ ↑ ↑ ↑ or [[[[	Causes the number to be printed in exponential (E or D) format. This will be displayed as a "[".

Computer programming I
Computer Literacy
Page 9
Evaluation

Name _____

Part I - True and False - Use a + for true, an 0 for false.

_____ 1.) Any statement that is in a program must be preceeded by a line number.

_____ 2.) NEW erases memory.

_____ 3.) 100 PRINT would give a space on the screen.

_____ 4.) REM staement allows you to place comments into a program which the computer will ignore when it executes a program.

_____ 5.) PRINT 10↑3 would print 30 on the screen.

_____ 6.) PRINT "10 + 6" would print 16 on the screen.

_____ 7.) The ; at the end of a print statement causes the computer to go to the next print line on the screen.

_____ 8.) If you type RUN, the computer gets up and runs away.

_____ 9.) The ; after a print statement causes variable to be printed close together.

_____ 10.) The line 30 GOTO 10 will cause the computer to goto line 30 of the program.

_____ 11.) 100 X = X + 1 is a good BASIC statement.

_____ 12.) The , after a print statement prints the value of the variables far apart.

_____ 13.) LIST causes the program to be erased from memory.

_____ 14.) The word READY on the screen means the computer does not understand what you have told it to do.

_____ 15.) When 10 INPUT Q is executed by the computer, a ⟩ symbol will appear on the screen.

_____ 16.) If you wish to start a program again you type the word RUN.

_____ 17.) System commands are commands that tell the computer to do something while under the control of the human operator.

_____ 18.) If you wish to branch to line 100 if x = 4, then you would use 50 IF X⟩ 4 THEN 100.

_____ 19.) If X = 1, then 1 would print on the screen when you executed PRINT "X".

_____ 20.) The FOR Next loop enables a series of program lines to be repeated easily without complex counting and comparison lines.

BEST COPY AVAILABLE

Using the following program, answer the questions below.

```
10 REM
20 PRINT "TYPE IN THE HIGHEST NUMBER YOU WISH TO PRINT";
30 INPUT N
40 X = 1
50 IF X > N THEN 100
60 PRINT X;
70 X=X+1
80 GOTO 50
100 END
```

21.) What does line 20 do?

22.) What does line 30 allow you to do?

23.) If N = 10 and x= 5, what line would be execute after line 50?

24.) If line 60 did not have the ; at its end, what would happen?

25.) What does the program do?

Questions 26-30: Name the five parts of any computer.(Answers were on video tapes.)

26.) _____

27.) _____

28.) _____

29.) _____

30.) _____

BEST COPY AVAILABLE

Explanation of Error Messages

NF NEXT without FOR. NEXT is used without a matching FOR statement. This error may also occur if NEXT *variable* statements are reversed in a nested loop.

SN Syntax Error: This usually is the result of incorrect punctuation, open parenthesis, an illegal character or a mis-spelled command.

RG RETURN without GOSUB: A RETURN statement was encountered before a matching GOSUB was executed.

OD Out of Data. A READ or INPUT # statement was executed with insufficient data available. DATA statement may have been left out or all data may have been read from tape or DATA.

FC Illegal Function Call: An attempt was made to execute an operation using an illegal parameter. Examples: square root of a negative argument, negative matrix dimension, negative or zero LOG arguments, etc. Or USR call without first POKEing the entry point.

OV Overflow: The magnitude of the number input or derived is too large for the Computer to handle. NOTE: There is no underflow error. Numbers smaller than ±1.701411E − 38 single precision or ±1.70141183454455 6E − 38 double precision are rounded to 0. See /0 below.

OM Out of Memory: All available memory has been used or reserved. This may occur with very large matrix dimensions, nested branches such as GOTO, GOSUB, and FOR-NEXT Loops.

UL Undefined Line: An attempt was made to refer or branch to a non-existent line.

BS Subscript out of Range: An attempt was made to assign a matrix element with a subscript beyond the DIMensioned range.

DD Redimensioned Array: An attempt was made to DIMension a matrix which had previously been dimensioned by DIM or by default statements. It is a good idea to put all dimension statements at the beginning of a program.

/0 Division by Zero: An attempt was made to use a value of zero in the denominator. NOTE: If you can't find an obvious division by zero check for division by numbers smaller than allowable ranges. See OV above and RANGES page 221.

ID Illegal Direct: The use of INPUT as a direct command.

TM Type Mismatch: An attempt was made to assign a non-string variable to a string or vice-versa.

OS Out of String Space: The amount of string space allocated was exceeded.

LS String Too Long: A string variable was assigned a string value which exceeded 255 characters in length.

ST String Formula Too Complex: A string operation was too complex to handle. Break up the operation into shorter steps.

CN Can't Continue: A CONT was issued at a point where no continuable program exists, e.g., after program was ENDed or EDITed.

NR No RESUME: End of program reached in error-trapping mode.

RW RESUME without ERROR: A RESUME was encountered before ON ERROR GOTO was executed.

UE Unprintable Error: An attempt was made to generate an error using an ERROR statement with an invalid code.

MO Missing Operand: An operation was attempted without providing one of the required operands.

BEST COPY AVAILABLE

FD Bad File Data: Data input from an external source (i.e., tape) was not correct or was in improper sequence, etc.

LS DISK BASIC only: An attempt was made to use a statement, function or command which is available only with the Disk System.

COMPUTER PROGRAMMING II

COURSE OUTLINE

Week 1 - a. Course requirements
b. Review of Computer I concepts
c. Explanation of how a Time-share system works
d. Explanation of parallel verse series data transmission
e. Explanation of various computer information codes.
f. Introduction to logical operands.

Week 3 - Project 1, Right Triangle problem (see attached)

Week 4 - Strings
a. Manipulations
b. String functions (LEN,LRFT$,MID$,RIGHT$,VAL$,CHR$,ASC)
c. Project 2 - see attached

Week 5 - a. Explanation of how Dean's list is calculated
b. Dean's list project - see attached
c. Test on the review,logical operands, and strings

Week 6-7 - a. File structures (disk)
b. Explanation of how to read from a disk file.
c. Project 4

Week 7 - Midterm test

Week 8-9 - a. Printing to files
b. Use of temporary holding files
c. Introduction of variable files ASSIGN statement
d. Project 5

Week 10-14 a. Explanation of specialized Hewlitt-Packard functions as applied to files
b. Introduction to basic bysiness terms concerning Inventory control
d. Dividing the class into three groups and do project 6 as though the class is a programming department of a large corporation.
b. Techniques of updating files.
Week 15-18 a. Explanation of basic business terms concerning payroll
b. project 7 - payroll

While working on projects 6 and 7 the class meets two days a week to learn basic FORTRAN concepts and how they relate to BASIC

Week 12 - Types of variables,designators, calculations
Week 13 - Format statements, reading, writing
Week 14 - If decisions - FORTRAN project 1
Week 15 - DO-CONTINUE statements - FORTRAN project 2
Week 16 - Arrays and matrices, test

BEST COPY AVAILABLE

COMPUTER PROGRAMMING III

COURSE OUTLINE

Week	Topic of Discussion
1	Review of payroll program from Computer Programming II
2	Introduction to graphics a.) SET, RESET, POINT, PRINT @ b.) Project 1 - design and draw a picture or design on the screen
3-5	Expansion of graphic techniques a.) use of CHR$, special characters, control characters b.) use of VAPRTR, string packing, POKE, etc. to enhance and speed up graphics movement and generalizations on the screen c.) Project 2 - develop a program to play a game that combines graphics and text on the screen simultaneously
6	Numerical Precision a.) Intger, regular, and double precision variables b.) DEFINT, DEFSNG,DEFDBL, and DEFSTR functions c.) Project 3 - write a program that would allow one to input checks includeng its number, payee, and amount, then print a check register with all necessary totals.
7-8	Sorting Techniques a.) study the Bubble, Shell, and Quik sorting routines b.) Project 4 - modify project 3 by adding an additional input for an account code, then sort the checks by account code and print the checks by account code. Print subtotals for each account.
9-10	Program package development a.) define what a program package is and what it can do for the user. b.) Determine what a good package would include - i.e. menu, editing, printer formats, etc. c.) Project 5 - modify project 4 to have it menu driven with at least 2 printout formats, 2 sort formats, and editting of checks.
11-12	Numerical Approximation a.) Approximatation of answers by numerical analysis techniques b.) Project 6 - write a program to find the area under a curve by either the trapezoidal or rectangular rules.
13-14	Sequential Files a.) Various types of file structure and access b.) Introduction of OPEN, CLOSE, INPUT #, PRINT #, EOF functions or commands c.) Project 7 - write a program to read a sequential file made up of a string followed by 4 double precision numbers, make various changes in selected fields, and place the changes back into the file.
15-17	Random Access Files a.) Introduction to FIELD,GET,PUT,LSET,RSET,CVD,CVI,CVS,MKI$,MKS$,MKD$,EOF, and LOC commands b.) File search techniques c.) Project 8 - Write a program to search in a random file for requested fields and print them on paper

29. A Pythagorean Triple is a set of three positive integers a, b, and c such that $a^2 + b^2 = c^2$. Use this information the next problem.

Find all Pythagorean Triples whose components are all less than or equal to fifty. Computer time can be saved by assuming $a < b < c$. (Note that if $a = b$, c cannot be an integer.) Then the integers to be tested can be generated by the following statements.

```
10 FØR A = 1 TØ 48
20 FØR B = A + 1 TØ 49
30 FØR C = B + 1 TØ 50
```

30. Print the rational numbers in lowest terms between zero and one which are of the form A/B where A and B are integers from 1 to 10, inclusive. The list will look like this.

1/2 1/3 1/4 1/5 1/6 1/7 1/8 1/9 1/10

2/3 2/4 2/5 2/6 2/7 2/8 2/9 2/10

... ...

8/9 8/10

9/10

31. Determine the limit of $\frac{\sin x}{x}$ as x approaches zero by printing x and $\frac{\sin x}{x}$ for x = 1, .5, .25, .125, . . .until the difference between successive values of the ratio is less than .00005.

32. Suppose that the world population is 3.7 billion. Each year it increases by 1.9%. Calculate the number of years from now in which world population will be double its present level and the number of years from now in which it will be triple its present level (assuming that the present growth rate remains constant).

33. Generate a positive integer, print its positive integral factors. Run 10 times.

34. A "palindromic prime" is a prime that is also a prime when its digits are reversed. 11, 13, and 17 are such primes. Find 10 others.

35. Find twelve consecutive composite integers. Do not use a trial-and-error method. (Hint: Find an algorithm that involves a factorial.)

36. Given the number of pins he knocked down with each ball in each frame, compute a bowler's score.

BEST COPY AVAILABLE

37. A function f is even if, for each x in the domain, $f(x) = f(-x)$. A function f is odd if $f(x) = -f(-x)$ for each x in the domain. Decide whether the function determined by P is even, odd, or neither. Input P's coefficients. Print the first derivative of the function determined by P. Print the second derivative.

38. For a cubic polynomial function, find the critical point(s), if any, and determine whether each is a relative maximum or relative minimum.

39. Use Newton's Method to approximate an irrational zero of a polynomial function, given the two consecutive integers between which the root lies. Newton's Method is based on the iteration formula.

$x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$ for $n = 1, 2, 3, \ldots$, where $f'(x_n)$ is the first derivative of f at x_n.

BEST COPY AVAILABLE

Comp. Math
Prog. 4
Mr. Kulp

Name _____

Level C Problems

1.) Write a program to calculate and print the reciprocals and squares for integers 1 to N where N is entered from the keyboard at the time of execution. Make sure your columns are headed correctly. Limit N to a m maximum of 10 when you run the program.

2.) A man is paid 1 cent the first day, 2 cents the second day, 4 cents the third day, etc. doubling each day on the job for 30 days. Write a program to calculate his wages on the 30th day and his total wages for the month. Label and print the results.

Level B Problems

3.) Do problem number 1 above and in addition to it find the sum of each column. After you print the table skip two lines and then print the sum of each column under its respective column.

4.) Write a program to calculate the amount in a savings account after a specific number of years. Write the program so that a person will input the interest rate as a percent, number of compoundings per year, original amount in the account, and the number of years you are to leave it in the account. Write the program so that it prints the amount in the account after each year.

Level A Problems

5.) Write a program that will print a table showing compound interest on 1000 dollars for 1,2,3,and 4 years at 4%, 4.5%, 5%, and 5.5% compounded quarterly and daily. Set up your own style of output so that a person not knowing much about the program can easily read the output.

6.) Write a program to read from a data block test scores and then find:

a.) the average score
b.) the number of A, B, C, D, and F's there were using

90 - 100	A
80 - 89	B
70 - 79	C
60 - 69	D
0 - 59	F

c.) Print the number of tests as well as the above.

BEST COPY AVAILABLE

Computer Programming I
Mr. Kulp
Project 5

Name _____

Level C Problems

1.) Write a program to convert inches to yards, feet, and inches given the number of inches from the keyboard or to convert yards, feet, and inches to inches. This is to be one program and the user is to choose which conversion he wants to do at the beginning of the program.

2.) Write a program to find all positive integers in the form $2^n + 1$ that are divisible by 3. Let n be inputed from the keyboard at the time of execution. (In your test runs limit the maximum size of n to 20.)

Level B Problems - Everyone

3.) Write a program to find and print all positive integer factors of a given positive integer inputted from the keyboard at the time of execution.

4.) A program to change an improper fraction to a mixed number. All fractions must be reduced to simpliest form.

5.) Write a program that would reverse the digits of a three digit number, print out this reversal, then determine whether the reversal is a factor or multiple of the original number.

Level A problems - For those students who have not studied trig.

6.) Write a program to find the roots of any quadratic equation, printing out both the type of root (rational, irrational, or complex) and the actual roots.

7.) Write a program to determine whether a number inputed from the keyboard is deficient, perfect, or neither.

Level A Problems - For those students who have studied trig.

8.) Write a program to solve a right triangle given any two parts of the triangle beyond the right angle. For example, you are given two legs, you are to find the hyp. and 2 angles, or you are given an angle and a opp. leg, you find the other angle, leg, and hyp., and so on for all possible combinations.

9.) Write a program to solve any oblique (non-right) triangle given any three parts. (Laws of Sines and Cosines)

BEST COPY AVAILABLE

```
1 REM PROGRAM TO PRINT GRAPH POINTS OF POLYNOMIAL
10 FOR J=1 TO 4
20 PRINT"INPUT COEFFICIENT";J;
30 INPUT C(J)
40 NEXT J
45 CLS
46 PRINT"EQUATION IS Y=";
50 FOR K= 1 TO 3
60 PRINT C(K)"* X ["(4-K)"+";
70 NEXT K
80 PRINT C(4)
85 PRINT
90 PRINT"X     Y"
100 FOR X=-5 TO 5
110 Y=0
120 FOR J= 1 TO 4
130 Y=C(J)*X[(4-J)+Y
140 NEXT J
150 PRINT X;Y
160 NEXT X
```

```
EQUATION IS Y= 2 * X [ 3 + 4 * X [ 2 + 5 * X [ 1 +-7

     Y
-5  -182
-4  -91
-3  -40
-2  -17
-1  -10
 0  -7
 1   4
 2   35
 3   98.0001
 4   205
 5   368
READY
>.
```

BEST COPY AVAILABLE

Computer I
Project 6
Mr. Kulp

Name _____

BEST COPY AVAILABLE

Option A - Do one of the following projects

NOTE: All projects must contain an array. The array must contain a minimum of 15 elements except for problem number 6.

C Level Problems

1.) Write a program to read in a list of numbers from a data block, print it in original order, then in reverse order. Write the program so that the first number in the data line is the total number of elements in the list.

2.) Read a list of numbers from a data block which contains both positive and negative integers. Place into array P the positive (and zeroes) and the negative numbers into array N. Then print the original list, the number of positive, the number of negatives, and the two arrays P and N. Write the program so that the first number in the data line is the total number of elements in the list.

B Level Problems

3.) The same as project 1 but in addition put the numbers in ascending order and then print this list.

4.) Write a program ot read a list of numbers; computer the average of the set; then determine how far from the average (above or below) each element in the list lies. Print the list with the difference from the avergge next to it.

5.) Write a program that will take 12 numbers from the keyboard at the time of execution and find the longest sequence of non-decreasing numbers . Print the original sequence and the longest sequence of non-decreasing numbers. For an array in which there are sequences of equal lengths, print them all. For example the list 17,31,40,5,6,10,46,37,53,12,9,86 has the longest string of non-decreasing numbers 5,6,10,46.

A Level Problems

6.) Write a program to input the coefficients of a polynominal function (min. of 3rd degree) and preform the synthetic dividion process from -5 to 5 by step 1 on the function. The output will be each coefficient of the quotient function and the remander in the form you learned in PreCalculus.

7.) Input the coordinates of the vertices of an n-gon (polygon with n sides) in the Cartesian plane. If the vertices are $(x_1,y_1),(x_2,y_2),(x_3,y_3)$, etc. input th total number of points first followed by each order pair which are placed into 2 arrays called x and y. Then find the perimeter and decide wnether the polygon is equilateral.

8.) The standard deviation of a list of data measures how "spread out" or "scattered" the data is. Specifically to find the standard deviation you do the following: a.) computer the mean (average), b.) subtract each item in the array from the mean and square this difference; c.) sum the squared differences of b; d.) divide this sum by the number of elements in the list; and e) take the square root of the quotient from d. Write a program to accept a list of data from a data block and print the list, number of elements in the list, the mean, and the standard deviation.

Option B - Write a program of your choosing using an array. Clear the project with the instructor befor proceeding;.

On the Radio Shack we have a 64 space x 16 space screen. The following are ways to it output at different locations on screen/paper. The ways are either line by line ontrols or individual line controls.

NE BY LINE CONTROL

ntrol	Meaning	Examples	Output
:(or LPRINT)	what follows will be printed on next line.	(#1) 10 PRINT 20 PRINT 4	 4
		(#2) 10 PRINT"PROBLEM#1" 20 PRINT 30 PRINT"PROBLEM#2"	Problem #1 Problem #2

NDIVIDUAL LINE CONTROLS

OTE: , ; and " " are called delimeters. A delimeter must be used between variables.

Control	Meaning	Examples	Output (16 spaces)
,	what follows will be printed at the start of the next zone (There are 4 zones on each line ⇒16 spaces to a zone)	(#1) 10 PRINT 3,7	3 7
		(#2) 10 PRINT 2,6 20 PRINT 3,8	2 6 3 8
		(#3) 10 PRINT"WORD"2,6	WORD 2 6

* Between literal printing and variable(s) or variable(s) and literal printing the delimeter " " is used.

;	what follows will be printed close to the preceding. see below*	(#1) 10 PRINT 2;9	2 9
		(#2) 10 PRINT 4,7;2	4 7 2
		(#3) 10 PRINT 2;6 20 PRINT 102;45	2 6 102 45
		(#4) 10 PRINT 2"THIS"8;7	2 THIS 8 7

* Remember every number has a sign before it and a space after

So, 17 = + 17 _

Then, 10 PRINT 4;5 = _4__5_

BEST COPY AVAILABLE

Control	Meaning	Examples	Output
(□) column number	what follows will be print in column number □		

space between TAB and (□)

NOTE: TAB(□) is considered to be an operation. A delimeter must be used between a variable and operation or operation and a variable A ; is generally used but a , also works.

16 spaces

8

column number 5

(#1) 10 PRINT TAB(5);8

(#2) 10 PRINT 6;TAB(21);4 — 6 4

(#3) 10 PRINT TAB(7);3,7 — 3 7

(#4) 10 PRINT TAB(18);4;8 — 4 8

NOTE: a delimeter must be used between operation and literal or literal and operation. " " are generally used.

3 PEN 7

(#5) 10 PRINT 3"PEN"TAB(10);7

WORD

(#6) 10 PRINT TAB(6)"WORD"

* The following are bad examples - why?

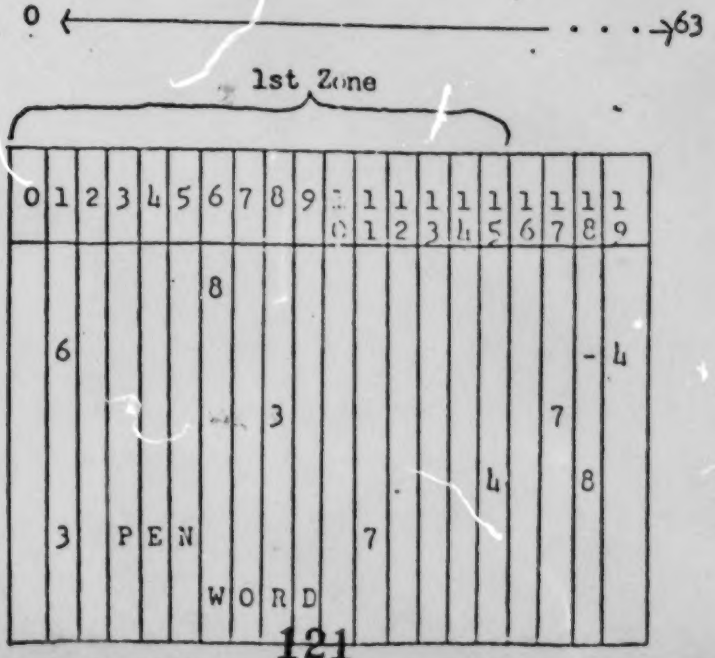
(#1) 10 PRINT 3,TAB(7);6 — 3

This will not work!

(#2) 10 PRINT 2;TAB(1);5; — 2 5 8
20 PRINT 8

This will work but is poor!

Note: print is sometimes used to overide , or ; at end of a line



BEST COPY AVAILABLE

Comp. Prog. I Project #7 Salesman Project

You are a salesperson from the Bucks County Plumbing Corporation dealing in 5 items having fixed prices. Suppose the 5 items are priced $1.10, $2.20, $0.80, $3.40, and $1.90 respectively. Write a program so that whenever you receive an order you can input the number ordered of each item and the computer will print the format given below for that order. Make the program practical for the person inputing as well as the person looking at the output.

Hint: Read in fixed prices of each item and use subscripted variables P(K) for price list and N(x) for the numbered order of each item.

BUCKS COUNTY PLUMBING CORPORATION

Salesperson: (Your Name)

— — — — — — — — — — — —

— — — — — — — — — — — —

Item #	Cost/Item	Number ordered	Cost
Sample 1	$1.10	3	$3.30

Total Cost $ [ ]

Grading

B - do all above

A - Do all above plus have all decimal points line up exactly

Computer Programming I
Project 8
Mr. Kulp

Name _____

Date Due _____

Part I

You are to construct a questionnaire of five (5) questions with five (5) weighted multiple choice responses as discussed in class. Write down the questions and have them approved by your instructor.

Part II.

After you have final approval of your questions, interview at least ten (10) people and record their responses below. (more if you wish)

Part III

Write a program to analyze the above information as taugh to you in class. The print out will be a chart which must include: a.) number of people responsing; b.) question number; c.) number of people who responsed to each question with their responses totaled; and d.) the weighted average for each question. Make sure everything is labeled and easily interpreted.

Sample printout (yours does not have to be an exact copy.)

Number of people responsing is 10

Question number Number	Response 1	2	3	4	5	Weighted Average
1	0	1	3	4	2	3.7
2	3	6	1	0	0	1.8
3	0	0	0	10	0	4
4	0	3	4	3	0	3
5	1	2	3.	2	2	3.2

Person	Response to question number 1	2	3	4	5
1					
2					
3					
4					
5					
6					
7					
8					
9					

BEST COPY AVAILABLE

2

```
 5 REM CARD DEALING PROGRAM
10 DIM C(13,4)
20  FOR Z= 1 TO 5
30 S=RND(4)
40 C= RND(13)
50 IF C(C,S)=1 THEN 30
60 C(C,S)=1
70 IF C>9 THEN 100
80 PRINT C+1"OF ";
90 GOTO 200
100 ON C-9 GOTO 110,130,150,170
110 PRINT "JACK OF ";
120 GOTO200
130 PRINT"QUEEN OF ";
140 GOTO 200
150 PRINT"KING OF ";
160 GOTO 200
170 PRINT"ACE OF ";
200 ON S GOTO 210,230,250,270
210 PRINT "SPADES"
220 GOTO 300
230 PRINT"CLUBS"
240 GOTO300
250 PRINT"DIAMONDS"
260 GOTO 300
270 PRINT"HEARTS"
300 NEXT Z
```

Computer Programming I
Project # 10
Iteration Process

Name _____

This project is designed to show how a computer can determine the answer to a problem through an approximation method.

Which proceedure you will do depends upon which mathematics course you are currently taking or have completed.

Procedure I - For those studnets taking Calculus, Pre-Calculus, or the Intermediate/ Pre-Calculus courses.

Determine the root of a function accurate to .0001 by Newton's Method. Be sure to put in a test for the first derivative equalling zero and pprint an appropriate message.

Formula: $x_{(new)} = x_{(old)} - f(x_{old})/f'(x_{old})$

Procedure II - All other students

Determine the root of an equation accurate to .0001 by the Bisection Method.

$x(new) = (x_{(upper)} - x_{(lower)})/2$

NOTE: error determination

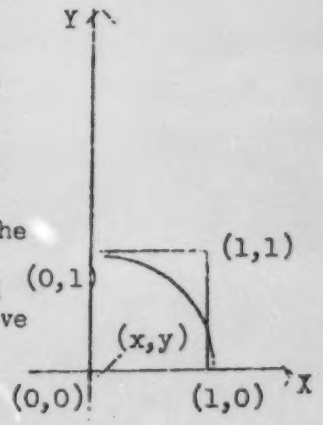
IF $ABS(x_{(new)} - x_{(old)}) < .0001$ Then XXX

Computer Programming I

Name _____

Date _____

1. A "Monte Carlo" method for calculating π goes as follows: consider a unit square in the xy-plane with vertices at (0, 0), (1, 0), (1, 1), and (0, 1). For a circle centered at the origin with radius one (see diagram at right), one quadrant lies within the unit square. The area of this circle is π, and therefore the area of the quadrant shown is $\frac{\pi}{4}$. Imagine randomly choosing random points within the square. Some of the points will lie within the circle and some will lie outside the circle. Because of the respective areas of the circle quadrant and the square, for a large number of points, approximately $\frac{\pi}{4}$: 1 of the points will lie within the circle. Write a program that generates random values for x and y which serve as the coordinates of a point within the square. The distance of this point from the origin is computed. If this radius is less than one, the point lies within the circle. If it is greater than one, it lies outside the circle. Count the number of times the point falls within the circle. The ratio of the number of points falling within the circle to the total number of points generated will be an approximation of $\frac{\pi}{4}$. Multiplying this ratio by four therefore yields an approximation of π itself.



1a. A magic square is an n x n array of the integers 1 to n^2 placed in such a way that the sum of each row, each column, and each diagonal is the same. See the example below.

2	9	4
7	5	3
6	1	8

Print a magic square of order n.

2. Prepare a checkbook-balancing program. Given the previous balance and the deposits and withdrawals for the month, print a list of the monthly transactions and the balance at the end of the month.

3. Program the computer to play blackjack against human opponents with the computer being the dealer.

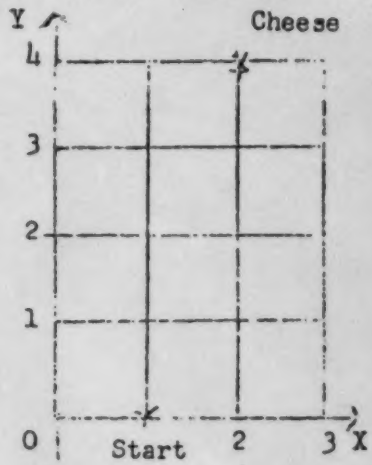
4. Deal and analyze a poker hand. Possible outputs include "pair of eights," "full house", and "2 - 6 straight".

5. A roulette wheel contains 38 numbers: 0 - 36, and 00. A player may bet on (a) a particular number; (b) even or odd; (c) red or black where 1, 3, 5, 7, 9, 12, 14, 16, 18, 19, 21, 23, 25, 27, 30, 32, 34, 36 are red, 2, 4, 6, 8, 10, 11, 13, 15, 17, 20, 22, 24, 26, 28, 29, 31, 33, 35 are black, and 0 and 00 are colorless; (d) the number falling in the first half (1 - 18) or the second half (19 - 36), and (e) the number falling in the low (1 - 12), middle (13 - 24), or high (25 - 36) range. Write a program to accept a number from 0 to 37 (37 represents 00) and print its characteristics (even-odd, red-black, 1st-2nd half, low-middle-high).

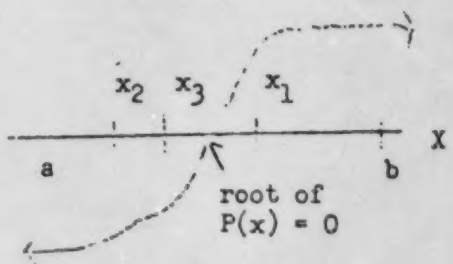
6. ~~Note: (this problem should follow No. 5)~~

Under the "Martingale" betting system, a player begins with a basic bet. Then when ever he wins, he repeats the basic bet. If he loses, his next bet is double the previous amount (unless his money is insufficient, in which case he bets the remainder of his funds). A sample sequence is the following: bet \$10 - win, bet \$10 - lose, bet \$20 - lose, bet \$40 - win, bet \$10 - lose, and so forth. The player continues until he either exhausts his capital or reaches a predetermined upper limit. Expand the roulette program to test this system.

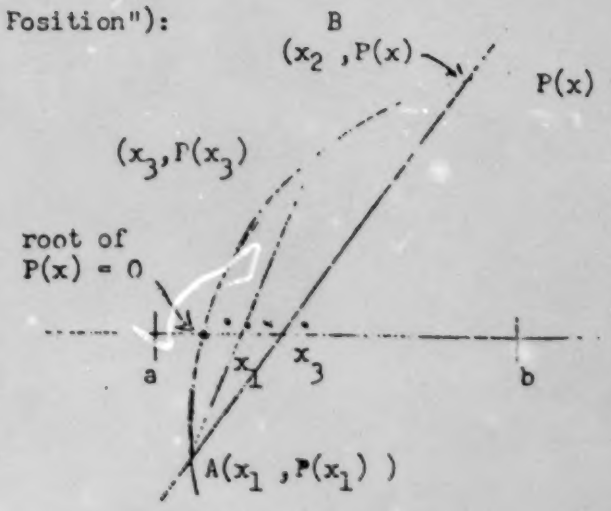
7. Simulate the trip of a mouse through a maze in search of cheese. In the diagram on the right, each intersection point of the maze has been given x and y coordinates. The mouse starts at (1, 0). At an interior intersection point such as (1, 1), assume that the probability the mouse goes in any direction is $\frac{1}{4}$. For a corner point such as (3, 0), assume that the probability is $\frac{1}{2}$ for either direction. For other border points such as (0, 2) the probability is $\frac{1}{3}$ for any direction. The mouse's trip ends when he reaches the cheese. Find the number of moves in each trip.



8. The Bisection Method: Suppose it is known (by substitution) that $P(a) < 0$ and $P(b) > 0$ (see the diagram at the right). Then P, being continuous, must cross the x-axis between a and b. To estimate the x - value of the crossing point, find the mid-point x_1 of the interval (a, b).
If $P(x_1) > 0$, the root lies in the left half of the interval (a, b); if $P(x_1) > 0$, the root lies in the right half (for the diagram shown).
In either case half the original interval can be discarded. Now bisect this new, smaller interval and repeat the analysis. By successive bisections the intervals in which the root is "trapped" become smaller and smaller and the root can be approximated to any desired accuracy.



9. The Method of Chords ("Method of False Position"): In the diagram at the right Chord AB crosses the x-axis "close" to the root of P(x). The slope of



AB is $\dfrac{P(x_2) - P(x_1)}{x_2 - x_1}$. By the point-slope form, the equation of AB is

$$y - P(x_1) = \frac{P(x_2) - P(x_1)}{x_2 - x_1} (x - x_1)$$

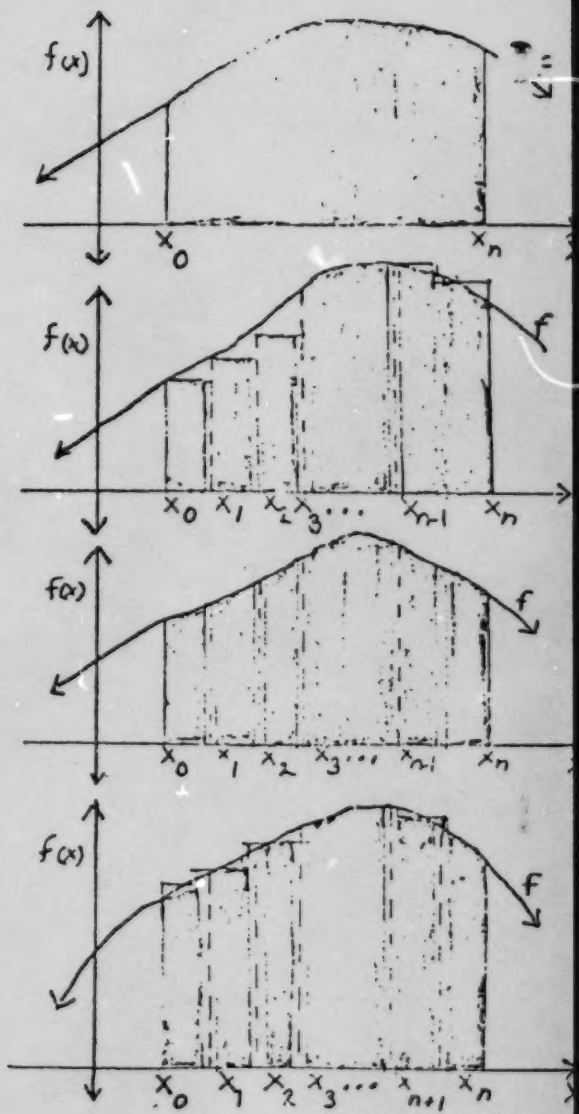
Substituting the coordinates of the point $(x_3, 0)$ into this equation and solving for x_3, we obtain

$$x_3 = x_1 + \frac{P(x_1)(x_1 - x_2)}{P(x_2) - P(x_1)} .$$

Now iterate the technique, first using the chord from $(x_1, P(x_1))$ to $(x_3, P(x_3))$ to gain a closer approximation x_4, then using the chord from $(x_1, P(x_1))$ to $(x_4, P(x_4))$ to gain a still closer apprximation x_5, and so forth. Write a program that uses the Method of Chords to approximate a zero of a polynomial function.

10. The "progressive" system of gambling can be used for any bet that is close to even-money (such as "even-odd" or "high-low" in roulette). The player begins with a sequence of numbers, say 1, 2, 3, 4, 5. For each play he bets the sum of the first and last terms ($6 for the first bet using the sample sequence). If he wins, he crosses off the first and last terms; if he loses, he appends the amount just bet to the end of the list. Thus if the first play wins, the sample list becomes 2, 3, 4 and the next bet is $6. If he loses, the list becomes 1, 2, 3, 4, 5, 6 and the next bet is $7. The sequence should eventually disappear, at which point (theoretically) the player has won the sum of his original sequence. Test this approach for roulette.

Briefly explained below are four methods for finding the area under a curve, as shaded in the figure on the right. In each case write a program to compute the area of a given function over a given interval by the method explained.



11. Rectangular Rule: Divide the interval (x_0 , x_n) into n equal parts and use each interval as the base of a rectangle that has one vertex on the curve f, as in the figure on the right. The sum of the areas of the rectangles approximates the total area under the curve. To obtain a better approximation, increase (for example, by doubling) the number of rectangles.

12. Trapezoidal Rule: This technique is similar to the Rectangular Rule except that trapezoids replace the rectangles (see the figure on the right). Again the approximation can be improved by increasing the number of trapezoids.

13. Midpoint Rule: Rectangles are used, as in Problem No. 13, but the midpoint of the top side of each rectangle rests on the curve f (see the figure on the right). Note that a portion of each rectangle lies above the curve to counterbalance the small areas under the curve that are not covered by the rectangles. (This last fact helps explain why this method gives a better approximation that either the rectangle or trapezoid rules.)

14. Simpson's Rule: This method, even more accurate than the Midpoint Rule, was invented by Thomas Simpson (1710-1761). The mathematics behind this method cannot be summarized in a diagram. Divide the interval (x_0 , x_n)

$Y_0, Y_1, Y_2, \ldots, Y_n$ where $Y_i = f(x_i)$. Then

area $= \frac{h}{3} (Y_0 + 4Y_1 + 2Y_2 + \ldots + 2Y_{n-2} + 4Y_{n-1} + Y)$ where h is the

width of interval (x_0 , x_1), (x_1 , x_2), . . ., (x_{n-1}, x_n). Again a

better approximation can be obtained by increasing n.

15. Generate the coordinates of a point in the xy-plane. Decide the number of the quadrant in which the point lies or, if it lies on an axis, decide whether it lies on the x-axis, on the y-axis, or on both.

16. Generate integer (1 to 20) numbers A, B, and C, which are the coefficients of a linear equation of the form $AX + BY = C$. (Assume that A and B are not both zero).

    (a) Compute and print the slope of the graph of the equation. Remember that the slope may be undefined.

    (b) Print the x-intercept of the graph as an ordered pair.

    (c) Print the y-intercept of the graph as an ordered pair.

    (d) Generate real numbers R and S. Decide whether the point (R,S) lies on the graph of $AX + BY = C$.

17. Generate 10 linear equations $AX + B = C$ (A, B, C real numbers), solve for x. For example, if the equation is $7x + 5 = 19$, print $x = 2$ and the equation.

18. Generate (1 to 10) linear equations of three points in the xy-plane, determine if the points are collinear. If the points are not collinear, decide whether the triangle formed by joining the three points is ISOSCELES, EQUILATERAL, or SCALENE. Make all print outs understandable.

19. A parabola is defined by $y = AX^2 + BX + C$. $A \neq 0$. Read rational numbers A, B, and C; 6 consecutive trials, and:

    (a) Print the coordinates of the vertex of the parabola.

    (b) Print whether the vertex is a maximum or minimum point.

    (c) Print the equation of the axis of symmetry.

    (d) Print the coordinates of the focus.

    (e) Print the equation of the directrix.

20. The "slugging percentage" of a baseball hitter is the ratio of his total bases to his at-bats. Generate his number of singles, doubles, triples, home runs, and at-bats, at random, and print a player's slugging percentage. Note: Like batting average, slugging percentage is expressed as a three placed decimal. A slugging percentage over .500 is considered outstanding. Run the program 10 times.

21. Find the area of a triangle, by generating the lengths of the three sides. Be sure to test to see if the lengths generated can make a triangle.

22. Generate a point P(x,y), print the ordered pair of the point which is symmetric to P with respect to the following. Run the program 10 times.

    (a) the x-axis

    (b) the y-axis

    (c) the origin

    (d) the line defined by y = x

23. Compute square roots of positive integers using Newton's Method, which consists of an alternation of dividing and averaging. Stop when the estimate squared is within .001 of the given number. Run 10 times.

24. Find the roots of any quadratic equation with rational coefficients. Run 5 times.

25. There are only two whole numbers less than 10,000 that are multiples of their reversals (where the reversal does not equal the original number). Both these integers are greater than 8700. The last digit of the original cannot = 6.

26. There are only two more integers less than 100,000 that are multiples of their reversals (again excluding the trivial case where the reversal equals the original number. These two integers lie between 87900 and 99000. Find them.

27. Find the number of distinct triangles with sides of integral length no greater than ten. If the lengths of the sides are A, B, and C, the program can be made more efficient by assuming $A \leq B \leq C$. Then these statements can be used to generate the integer combinations to be tested.

```
50 FØR A = 1 TØ 10
60 FØR B = A TØ 10
70 FØR C = B TØ 10
```

    Using the above conditions, count and print those combinations where A, B, and C are relatively prime (that is, one is their only common factors.)

28. Approximate π by finding the perimeters of regular n-gons inscribed in a circle of diameter one for n = 4, 8, 16, 32, 64, 128, 256.

Microcomputers in Engineering Graphics

William H. Park
Professor ofMechanical Engineering
Penn State University

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

MICROCOMPUTERS IN ENGINEERING GRAPHICS

William H. Park
Professor of Mechanical Engineering
Penn State University

Prepared for the Second Annual Microcomputer Information Exchange Conference, Penn State Campus, March 11-12, 1983

NETWORK DESCRIPTION

In February 1983, a network of Apple microcomputers was installed for use in Engineering Graphics EG 50, a course taken by most engineering freshmen.

The network used is the Remote Operating System (ROS) by Softworks, Scottsdale, AZ. At the time of system purchase, this was the only available networking system for Apples. The ROS system allows multiple Apple computers to share the same disk storage available from one CENTRAL computer. This provides each REMOTE the flexibility as though a disk drive were directly connected, without the costs associated in purchasing a separate disk drive for each computer. The hardware consists of a central card placed in any slot except 0 on the CENTRAL computer and remote cards in slot 7 of each REMOTE computer. The cards are then cabled together in serial fashion using a ten wire flat ribbon cable. Up to 127 remotes can be networked. The CENTRAL computer supports up to 8 floppy discs.

After installation of the hardware, DOS 3.3 and the Remote Operating System AROS is loaded into the central computer from the systems disk supplied. A hybrid DOS/AROS operating system is loaded into each remote. Simultaneous BOOTing of multiple remotes can be performed automatically. All DOS commands except INIT are executable by each remote.

A PASSWORD table must be established before any remote is allowed to log onto the network. The PASSWORD table may contain up to 500 unique passwords of which no more than 127 may be active on the ROS network at any one time. Each password contains the Password Code, Name of user, Tag, Reference, and Drive Configuration.

TURNKEY systems may be implemented allowing a specific DOS command and an optional program name to be executed automatically when a remote logs onto the network.

PUBLIC is a program which allows a remote to make a private file public and thus accessible by any remote user.

In addition to DOS commands, AROS supports several special commands:

- FEED allows a program, defined at the CENTRAL computer, to be simultaneously loaded to multiple remotes.
- KAT redisplays the last catalog a remote executed without accessing the disk drive.
- BYE allows a remote user to log off the network.

BEST COPY AVAILABLE

Our network has only been operational since early February. In preliminary work with six sections of students, no serious problems have been encountered, except for disk storage volume. We know that for more than two or three sections of students, a hard disk would be needed. For spring term, a Corona 10 meg disk, which is to arrive next week, will be installed.

Printing by each remote is done through the central print spooler program which dumps files to a remote computer dedicated to printing. The printer is an Epson MX80 with 32k Microbuffer. The print spooler is also by Softworks and had not been released when we obtained a copy. A few problems have been encountered, but we were fortunate to get a copy of this version 0.0. We have been able to tell Softworks about these few problems and they will be corrected in the new soon to be marketed marketed version.

Hardware in the Laboratory consists of:

17 Apple II+ computers, as REMOTES
1 Apple II+ computer, 4 disks as CENTRAL
1 Apple II+ computer, printer as PRINT SPOOLER
1 Apple II+ computer, disk as STAND ALONE
1 HP 7470A plotter

MICROCOMPUTER USE IN THE COURSE

EG 50 is a combination of engineering graphics and an instrumentation laboratory. The computer is being used in four areas of the course and consists of the following topics:

1. Computer Literacy
2. Graphical Visualization
3. Descriptive Geometry
4. Experimental Data Collection and Manipulation

The computer literacy consists of nine hours of activity related to an introduction to microcomputers and to programming. Commands using Applesoft Basic are covered along with data inputs, editing and formatting. Both low and high resolution graphics are introduced. The students are required to write various simple Basic programs. A self-paced instruction manual has been developed to challenge the typical engineering freshman.

The graphical visualization program was written by one of my senior students as an honors project. A pictorial drawing of an object is placed on the monitor, and the student is asked to sketch (on paper) the top, front and right side views of the object. They can then 'ask' the computer for the correct views for comparison to their sketches. One then can then return to the pictorial for review, or go on to another one. All data for the pictorial and its views are loaded into the Apple via an Apple graphics tablet. New objects can easily be added to the data base in this manner.

BEST COPY AVAILABLE

The descriptive geometry programs that have been written to date show true lengths of lines and true sizes of planes in three-dimensional space. Another program works out the distance between a point and a plane described by three points. The last program shows the piercing point of a line, or its intersection, and a plane. Various programs are available for this type of calculation, but most are done in "background" with no student input. These programs were written to solve the problems by using auxiliary views. This technique is identical to the way that the student learns to solve the problems using drafting equipment. The program is somewhat interactive in that the student must make several decisions during the solution.

The microcomputers are also being used in the instrumentation laboratory. Several experiments are designed to show the variability of data, and require that some statistical analysis be performed. Several Apples have been used for over a year in this phase of the program to calculate means and standard deviations, and to plot histograms of the data. Starting in the fall, we will also be using the apples to collect data from several types of transducers using analog to digital convertors. The data can then be manipulated and/or plotted for use in the student reports.

The reaction to this infusion of microcomputers into ED 82 has been very positive. It is planned that all freshman engineering students will be exposed to this activity starting next fall.

BEST COPY AVAILABLE

Computer Camps for Kids

Ms. Joan L. Parrett
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

SECOND ANNUAL MICROCOMPUTER

INFORMATION EXCHANGE CONFERENCE

Computer Camps for Children and Adults in America

by

Joan L. Parrett

Tennis, soccor, ice hockey, and football all have their own grand scale of summer camps, ranging from short-day sessions to posh residential ten-week camps, but a challenging stranger is presently threatening their dominance-computer camps. In the past four years, since their inception and especially during 1980 and 1981, computer camps have sprung up in several locations and with different opportunities for children and adults. The novelty, practicality, and aura surrounding them attracts thousands of applications for the five major camps each eight-week summer session. Campers come with various backgrounds, from no knowledge to those toting their own personal machines to insure instant access instead of the common method of sharing one for two children. On visitation days, parents and relatives are so frequently overwhelmed with their children's progress, that several camps offer special adult weekends, and others are beginning to cater only to the adult clientele. Common practice involves LOGO or BASIC language introduction, progressing only with the talents and interests of the camper. Instructors work directly with the camper or in small groups, being college computer science majors or certified teachers. The common goals

expressed by most of the camps are, "learning to think logically," (Dentzer, 1982), and instilling a sense of excitement and motivation.

The National Computer Camp, started by Dr. Michael Zabinski in July 1978 and still in operation, was one of the first. The Orange, Connecticut, camp ran from 9 to 4:30 daily. "The purpose of the camp was to provide youngsters with ample opportunities to use the computer for instructional as well as recreational applications. In the process, they came to understand the potential as well as the pitfalls involved in using computers" (Zabinski, 1979). This early experiment definitely foresaw the future of computers relegated only to the schools, rather than the home. The camp itself, was operated in the local junior high, which they described as "the appropriate academic setting" and "as providing the basis for subsequent study in a school setting" (Zabinski, 1979). All grades of children from the sixth through the twelth grades could apply during 21 two-week sessions. Eight Wang computers, cassettes and CRT micros, two printers, a card reader and a dual diskette drive unit comprised the camp's hardware. Again, the instructors were working with only two or three children on one level. The broad goals Dr. Zabinski hoped to achieve also involved dispelling the mystique surrounding computers during the 1970's, consideration of social responsibilities, and actual computer expertise in BASIC, algorithms, flowcharting, and modeling. The formal instruction and hands-on time was punctuated with recreational activities, required of each child with optional field trips, speakers and computer-related reading and research encouraged. This general format still supplies the common circuitry for modern computer camps.

The nation's largest camp is Computer Camp, Inc., started in 1980 by Denison Bollay in Santa Barbara, California, with his LSI (large-scale integration) camping program similar to Dr. Zabinski's. Mr. Bollary was an independent researcher and design consultant in Silicon Valley. While maintaining an excellent level of academic expectations and corresponding achievement with the campers, plus unusual optional activities, Mr. Bollay and Director, Garry White, demand that daily recreational activities be the other highly important aspect of their experience.

> If children want only to learn more about computers, Garry White suggests that they attend any of the many summer schools that concentrate on computer studies. Computer Camp's goals are broader. We want to get the children out of the classroom and into an environment that is more fun and creative. Studying the way our ecosystems work prepares kids for a world of calculators and computers by really going into the meat of it. By studying nature, our kids are getting the big picture of things. Playing rewarding physical games and being grouped together on special projects helps the children grow socially. (Cantillon, 1982)

"We stress that good programmers need healthy minds and healthy bodies" (Dentzer, 1982). Each child must sign-up for two daily recreational, traditional camp activities, along with three ninety-minute classes of instruction. For beginners, the classes cover BASIC and introductory programming. Intermediates cover advanced BASIC and introductory PASCAL, plus catalog functioning, ABS math calls, programming and group projects. The advanced students work on robotics, graphics, computer-generated speech and modular programming. After several days of intensive instruction, each child is expected to begin working on an individual project. The Softalk magazine and the Pacific Computer Company fund two scholarships annually to this camp.

Heuristic concepts presented by computer experts Leuhrmann, Bork, Papert, and Dwyer can be seen in the Computer Camp's philosophy and curriculum. "In the future," Bollay says, "those who are not computer-literate will be at a terrible disadvantage. Training kids to work with computers is like training them to read--it's that critical" (Pierce, 1981). Mr. Leuhrmann argues for national computer literacy too. Newsweek's Susan Dentzer reports that the campers at Santa Barbara are "quickly urged to abandon passive game-playing in favor of using the computer as a problem-solving tool" (1982). Bork utilized dialog and interaction to encourage such activity as problem-solving too. Papert's heuristic approaches can be seen in these following two quotations concerning the Camp's curriculum: "'You can learn interesting things while you're having fun!' exclaims one camper" (Dentzer, 1982). Another camper expressed: "'I want to make a CSI (Computer System Instructional Program) for teaching kids like us how to use some graphics and key words'" (Pierce, 1981). Dr. Arthur Michals the Director of Computer Camp International, formally Computer Camp East, first employed Papert's LOGO with the disabled and it proved to be an effective doorway for campers also.

> I became very interested in the work of Seymour Papert and his developments of articifical intelligence and began looking into the potential of using LOGO with the language-impaired and handicapped. Through LOGO and artifical intelligence, many learning-disabled people get a chance to prove themselves. It was when I understood this that I began to use computers in our programs. (Cantillon, 1982)

Dr. Michals believes their's is one of the only schools which places strong emphasis on computer ethics and morality.

With ever-increasing popularity of computing and the spreading belief in their essence to our society, the following attitude may be the cause of the relative slowness in the growth of computer camping: "A computer can't take the place of swimming and obstacle courses," (Rogers, 1982), a common attitude prevalent among traditional camp directors. Consequently, most camps nationwide have incorporated a curriculum of traditional camping activities. The cost of hardware is also an important consideration. Software libraries, available at only a few camps presently, offer greater horizons for the camper, but add tremendous overhead expenses. June Rogers reports further in Macleans that Canadian children interested in computers frequent American camps as only a few facilities exist in Canada. Foreign students increasingly comprise the annual list of campers. The map, showing the camps' locations, illustrates that the majority correspond to major metropolitan centers and university towns. Other camp directors believe computing skills are entirely the school's responsibility. Consequently, schools and summer camps for educators have increased in number and popularity.

The Wisconsin Department of Instruction, using ESEA Title IV-C Project funds, backed a similar project in Mukwonago, Wisconsin. Held at the local high school, 26 volunteering secondary and elementary teachers from the district were involved in two weeks of hands-on computing with a $12.50 daily stipend. The objectives were general information and computing skills, increasing teacher awareness of computer potential in their individual classrooms, and beginning programs for their special interest subjects. Although the directors felt the 80 hours of hands-on time was insufficient for more than the bare essentials,

overall feedback reflected very positive attitudes. The address for future contact is: Mr. Michael A. Burke, Coordinator of Instructional Resources, Mukwonago Area Schools, 423 Division Street, Mukwonago, WI. Upon returning to the classroom, the teachers reported identical student problems as the camp directors and instructors. These can be seen in the reactions I have overheard in the campus computer lab also: "Just one more time, I know it will work - I'll eat later." "Don't erase it! I'll be right back!" And "Golly, my fingers hurt!"

Whether in the classroom or at a summer camp, both young and old students find an irrestible challenge in computing, as well as certain predictable addiction. As computer scientists conclude, computer literacy will be and, in fact, is a necessity. Those now skilled are literally, off-and-running with the ball. The question remains, as Dr. Weizenbaum fears, will the necessity of a computer-centered society become, in the grand rush, synonymous with the general skill of computer literacy? And indeed, should it?

REFERENCES

Bueke, Michael, A., & Kaiser, Jerry. Who took a byte from my Apple ROM? National Association of Secondary School Principals' Bulletin, September 1982, 66, 83-86.

Cantillon, Jane. How the old campgrounds have changed. Softalk, May 1982, 2, 194-197.

Dentzer, Susan, & Friendly, David T. Computer camps for kids. Newsweek, July 19, 1982, 100, 72; 75.

Dunn, D. H. Summer computer programs for kids. Business Week, May 10, 1982, pp. 54.

Garris, Barbara. Summer computer programs. Creative Computing, May, 1982, 8 (5), 91.

Hiraki, Joan. How I spent my summer vacation: summer computer camps across the US. Educational Computer Magazine, May/June 1982, 2(3), 32-36.

Morris, Holly. Lures of summer school: a calculating program. Newsweek, August 3, 1981, 98 (5), 77.

Pierce, Kenneth M. Camps for computers. Time, August 3, 1981, 113, 70.

Rogers, June. Computers in the pines. Macleans, May 17, 1982, 95, 54.

Tommervik, Kirin, & Wahlner, Kurt. The newest face at summer camp is Apple's. Softalk, April 1981, 1, 72; 2.

Zabinski, Tobi F., & Zabinski, Michael P. Computer summer camps. Audio-visual Instruction, November 1979, 24, 35-36.

1982 Camps. The New York Times Magazine, March 14, 1982, p. 113-122.

1982 Camps. The New York Times Magazine, May 2, 1982, p. 108-116.

SECOND ANNUAL MICROCOMPUTER
INFORMATION EXCHANGE CONFERENCE

Computer Camps for Children and Adults in America

COMPARISON CHARTS OF SELECTED COMPUTER CAMPS

The following data was gathered from 1983 brochures except where noted by a ★ preceding the camp name. X denotes yes, and 0 meaning no. NA shows that the information concerning this aspect was not available.

Many new computer camps are being established yearly across the country and in Canada. Consequently, the following comparison is not exhaustive, but rather focuses on the larger camps, those in the Northeast and some which offer unusual curricula. The addresses and camp locations, some camping organizations offer several locations, are listed after the comparison sheets. The name of the current director(s) is listed also.

by

Joan L. Parrett

	A. GROUPING					B. HOURS per DAY				C.								
	BEGINNERS	INTERMEDIATE	ADVANCED	INDIVIDUAL INSTRUCTION	LARGE GROUP	FORMAL LECTURE INSTRUCTION	ASSIGNED GROUP WK.	ASSIGNED ID. WORK	INDIVIDUAL COMP. FREE TIME	HOURS per DAY	OPTIONAL PARTICIPATION	REQUIRED PARTICIPATION	TEAM SPORTS	INDIVIDUAL SPORTS	ARTS and CRAFTS	THEATER & DRAMA	ENVIRONMENT and CONSERVATION	FIELD TRIPS
ARIZONA COMP.CAMP	X	X	O	O	X	3	X	X	X	NA	O	X	X	X	O	O	XX	O
ATARI	X	X	X	X	X	1½	O	1-3	4-6	3	O	X	X	X	X	O	O	O
CHAMPLAIN COLLEGE	O	O	O	X	O	2	O	2	2-6	4-6	X	O	X	X	X	O	O	X
COMPUTER CAMP, INC	X	X	X	X	X	1½	←	3	→	1+	4	O	X	X	X	X	X	X
COMPUTER CAMPS INTERNATIONAL	O	O	O	X	O	X	X	X	X	NA	O	X	X	X	X	O	O	X
COMPUTER DAY C. AT DOBBS	O	O	O	X	X	2½	←	2½	→	X	2	O	X	X	X	X	X	O
DUCK	X	O	X	X	X	X	X	X	X	NA	O	X	X	X	X	X	O	X
4-H AT PURDUE	X	O	X	O	X	1	1½	3½	6	NA	O	X	X	X	O	O	O	X
HILL SCHOOL	X	O	X	X	X	3	O	←	6-8	→	3	X	O	X	X	O	O	O
MARIST	O	O	O	X	O	X	O	X	X	NA	X	O	X	X	O	O	O	X
NATIONAL C.C.	X	X	NA	O	X	←		12		→	NA	X	O	X	X	NA	NA	NA
PSU COMPUTER C.	X	X	O	X	X	3	O	←	3	→	O	X	O	O	X	O	O	O
TIMBERTECH	O	O	O	X	O	4	X	X	X	NA	X	O	X	X	X	O	O	O
[illegible]						11½		←	1	19½	→	O-	v	c	v	x	x	x

BEST COPY AVAILABLE

	D. CAMPER			E. LENGTH OF STAY and COSTS										F. STAFF						X
	AGES	EXPERIENCE REQUIRED	DESIGNATED WEEKS per AGE LEVEL	1 week		2 weeks		month		.8 wks.		extra weeks		COUNSELOR/CAMPER RATIO	COMPUTER BACKGRND. REQUIRED for COUNS.	RESIDENT COMPUTER EXPERT	CERTIFIED TEACHERS	OUTSIDE CONSULTANTS	SPECIAL SPEAKERS	ADULT CAMP OFFERED WEEKS or WEEKENDS
				AVAILABLE	COST	AVAILABLE	COST	AVAILABLE	COST	AVAILABLE	COST	AVAILABLE	COST							
RIZONA COMP.CAMP	8-14	O	X	X	260.	O	C	O	O	O	O	O	O	1:9	O	X	X	C	O	X
CARI	10-16	O	O	O	O	X	890.	X	1690.	X	2950.	X	425.	NA	X	X	X	X	X	O
HAMPLAIN COLLEGE	10-16	O	O	O	O	X	750.	X	1450.	6	2150	O	O	1:5	X	X	X	NA	O	X
OMPUTER CAMP, INC.	7-16	O	O	O	O	X	795.-1200.	X	1590. to 2400.	O	O	O	O	1:6	O	NA	X	NA	X	O
OMPUTER CAMPS NTERNATIONAL	8-17	O	O	O	O	X	835.	X	1650.	O	O	O	O	1:5	NA	X	X	X	X	X
OMPUTER DAY CAMP F DOBBS	4-16	O	O	O	O	O	O	X	575.	X	1045.	O	O	NA	NA	X	X	O	O	O
UCK	12-17	O	X	O	O	X	575.	O	O	O	O	O	O	1:15	X	X	X	X	X	O
-H AT PURDUE	11-18	O	X	X	250.	O	O	O	O	O	O	O	O	NA	O	X	X	O	X	O
ILL SCHOOL	NA	O	X	X	195.	O	O	O	O	O	O	X	195.	NA	NA	NA	NA	NA	NA	X
ARIST	10-17	O	O	O	O	X	795.	X	1590.	X	3180	O	O	1:10	O	X	X	O	O	O
ATIONAL COMP.CAMP	10-18	O	NA	X	345	X	690.	X	1380.	O	O	O	O	NA	NA	NA	NA	NA	NA	O
SU COMPUTER CAMP	12-17	O	O	X	385.	O	O	O	O	O	O	X	385.	1:20	X	X	X	O	C	O
IMBERTECH	8-17	O	O	X	495.	X	895.	O	O	O	O	O	O	NA	O	X	X	X	X	X
	12-						1115	X			250									

BEST COPY AVAILABLE

	G. LANGUAGES		H. COMPUTERS		I. COMPUTER CURRICULUM OFFERED EACH SESSION												
	1. BASIC 2. PILOT 3. LOGO 4. PASCAL 5. APL 6. FORTRAN 7. LISP 8. FORTH 9. machine 10. ASSEMBLER 11. COBAL	PRIMARY LANGUAGE TAUGHT	1. Apple 2. Atari 3. Commodore 4. Radio Sh. 5. Digital 6. IBM 7. TI 8. Sinclar or Vic 9. mainframe	CAMPERS to COMPUTERS RATIO	WORD PROCESSOR	DATA/FILE MANAGEMENT	SIMULATIONS/GAMES	ELECTRONIC WORKSHEETS	MUSIC/SOUND EFFECTS	MAILING LIST PROGRAM	GRAPHICS-COLOR	GRAPHICS-CHARACTER	GRAPHICS-ANIMATION	REQUIRED SECOND COMPUTER LANGUAGE	DAILY LECTURES	INDEPENDENT STUDY	GENERAL PROGRAMMING
ARIZONA COMP. CAMP	1	1	NA	1:1	O	O	X	O	O	O	X	O	O	O	X	X	X
ATARI	1,2,4,7,8,9,10	2	2	2:1	X	X	X	X	X	X	X	X	X	O	X	X	X
CHAMPLAIN COLLEGE	1,4,6,10	1	3,6,9	1:1	O	X	X	O	X	O	X	O	O	O	X	X	X
COMPUTER CAMP, INC	1,3,4,7	3	1,2,3,7	2:1	X	X	X	X	X	X	X	X	X	X	X	X	X
COMPUTER CAMPS INTERNATIONAL	1,2,3,4,5,6	1	1,3,7	1:1	X	X	X	O	X	O	X	X	O	X	X	X	X
COMPUTER DAY CAMP AT DOBBS	1,3,4,5,6	1	NA	2:1	O	O	O	O	O	O	X	O	O	X	X	X	X
DUCK	1,4,6	1	6,9	NA	← OPTIONAL →									X	X	X	X
4-H AT PURDUE	1,9,10	1	1	2:1	O	O	X	O	O	O	O	O	O	O	X	X	X
HILL SCHOOL	1,4,6	1	1,2,3,4,7,8 PDP 11/44	1:1	NA	NA	NA	NA	NA	NA	NA	NA	NA	O	X	X	X
MARIST	1,3,5	1	1,3,4,6,9	NA	O	O	X	O	X	O	X	X	X	O	X	X	X
NATIONAL C.C.	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
PSU COMPUTER CAMP	1,3	1	1	2:1	← OPTIONAL →									X	X	X	X
TIMBERTECH	1,2,3,4,10	NA	1	1:1 2:1	X	X	X	X	X	X	O	X	X	O	X	X	X
WESLEYAN	1,4,10,11	NA	1,2,3,4,5,6,7,8,9,11		X	X	X	X	X	X	X	X	X		X	X	X

BEST COPY AVAILABLE

Camp Addresses	Camp Locations
1. Arizona Computer Camp Dr. Henry Dahlberg 2946 Darca Drive Prescott, AZ 86301 (602) 445-3778	Mingus Springs Camp
2. Atari Computer Camps Dept. AL 40 East 34th Street New York, NY 10016 (212) 889-5200 (800) 847-4180	University of San Diego San Diego, CA University of North Carolina Asheville, NC East Stroudsburg State College East Stroudsburg, PA Stoneleigh-Burnham School Greenfield, Mass Oldfields School, Glencoe, MD Atheniam School, Danville, CA
3. Champlain College Computer Camp Shelly Richardson CCCC Room 28 163 S. Willard St. P. O. Box 670 Burlington, VT 05402 (802) 658-0800	on campus
4. Computer Camp, Inc. 1235 Coast Village Rd. Suite G Santa Barbara, CA 93108 (805) 969-7871 (800) 235-6965	England Tour Camp Santa Barbara, CA Lake Tahoe, NV Waimea, Hawaii Lake Sequoia, CA Winnipesaukee, NH Steamboat Springs, CO
5. Computer Camps, International (old name: Computer Camps East) Dr. Arthur Michals Suite D 310 Hartford Turnpike Vernon, CT 06066 (203) 871-9227	Banner Lodge, East Haddon, CT University of Wisconsin Whitewater, WI Selwyn School, Tenton, TX St. Mary's Hall, San Antonio, TX Oxford College, Oxford, GA
6. Computer Day Camp - REACH Mr. Marlin Fiedler Mr. Jim Finger 49 Clinton Avenue Dobbs Ferry, NY 10522 (914) 693-1400	Dobbs, The Masters School

7. DUCK — on campus
Duke University Computer Kamp
107 Bivins Blvd.
Duke University
Durham, NC 27708
(919) 684-6259

8. 4-H Microcomputer Workshop
Mrs. Helen Booriendi
AGAD Building, 4-H Dept.
Purdue University
West Lafayette, IN 47907
(317) 494-8426

9. Hill School Computer Camp
Mr. John E. Parnell
Pottsville, PA 17901
(215) 323-0823

10. Marist College Summer Computer Camp — on campus
Dr. Lawrence W. Menapace
Poughkeepsie, NY 12601
(914) 417-3240 (Ext. 345)

11. National Computer Camp — Westminster Prep. School, Simsbury, CT; Woodward Academy, Atlanta, GA
Dr. Michael Zabinski
P. O. Box 624
Orange, CT 06477
(203) 795-3049

12. PSU Summer Computer Camp — on campus
Room 410
Keller Conference Bldg.
The Pennsylvania State University
University Park, PA 16802
(814) 865-7557

13. Timber Tech Computer Camp — private camp
Mr. Nolan Bushnell
1287 Lawrence Station Rd.
Sunnyvale, CA 94086
(408) 745-1110

14. Wesleyan Summer Camp Institute — on campus at Middletown, CT
Mr. Russ Waller
92 St. Botolph Street
Suite 3
Boston, Mass 02116
(617) 266-8128

Parallel Microprocessors: Mainframe Power for Micro Prices

Mr. Douglas J. Parsons
Nuclear Engineering Department
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

Parallel Microprocessors: Mainframe Power for Micro Prices

Presented at: Second Annual Microcomputer Information Exchange Conference

Keller Conference Center
The Pennsylvania State University
University Park, PA 16802

Mr. Douglas J. Parsons
Nuclear Engineering Department
The Pennsylvania State University
(814) 865-1342

12 March 1983

Parallel Microprocessors: Mainframe Power for Micro Prices

As the title of this paper indicates, it is possible to combine several microcomputers operating concurrently in a parallel configuration for problem solving. In fact, there are specific classes of problems where the implementation of such a scheme becomes much more efficient than using a mainframe computer. I do not mean to imply that we do not need the mainframe, only that we should carefully consider each application and determine which computer size is best suited to our particular application.

There is a historic division of labor among the different sizes of computers, although the boundaries become more and more blurred as the technology in the field increases by leaps and bounds. The current distribution is briefly shown as follows:

Mainframes
- Massive computations ("number crunching")
- Large number of users, students/classes (i.e. a University environment)
- Large codes (programs) requiring significant amounts of main and/or auxiliary memory and greater accuracy.

Microcomputers
- Personal use (home, school, cottage industries)
- Programming instruction (learning BASIC, LOGO, etc.)
- Small business applications (spreadsheets, accounting, telecommunications)
- Devoted applications (data monitoring and collection, process control)
- Smaller codes requiring less storage and less accuracy.

Minicomputers
- Medium to large business applications
- Devoted scientific calculations
- Real-time simulations of large systems.

The natural overlap of the application of these technologies is greatly increased by the use of computer networks. A computer network is simply a group of two or more computers operating together. These machines may be located near one another and share a common central memory (i.e. a "tightly" coupled network) or they may merely be able to communicate with one another and transfer information over a large physical distance (i.e. a "loosely" coupled network). The computers in the network may be

mainframes, minicomputers, microcomputers, or any combination of these machines.

Under the heading of tightly coupled networks is a subgroup called "multiprocessors". The ANSI Vocabulary of Information Processing defines a multiprocessor as a system composed of two or more processing units under integral control. A more restrictive definition often applied by those in the field includes the simultaneous processing of two or more parts of the same program by two or more processing units under integral control. Currently, loosely coupled networks do not normally process different parts of the same program simultaneously and are generally not considered true multiprocessors.

In this paper I will describe an implementation of a loosely coupled network of microcomputers which is used in a true multiprocessing mode. Such a hybrid application is probably more aptly called a "multi-microprocessing ensemble". Figure 1 illustrates this configuration. The basic system is composed of several stand-alone APPLE II+ microcomputers (48K RAM) linked by a network system called the APPLE Remote Operating System (AROS*). The AROS consists of hardware and software which allows the APPLEs to share resources such as disk drives (e.g. disk files) and printers. As many as 127 APPLEs may share up to eight (8) common disk drives. For this project, I used eighteen (18) APPLEs sharing a single disk drive.

An effective measure of overall system performance is system throughput. Ideally, a multiprocessor with "N" processors operating at the same time should perform "N" times "better" than a single processor. It is usually not possible to obtain such an ideal performance for two reasons:

---

* Softworks, Inc.
14805 N. 73rd Street, Suite J
Scottsdale, AR 85260 (602) 998-3986

(1) The amount of parallelism is not uniform, that is, all "N" processors are not kept busy at all times.

(2) System performance is degraded as a result of system resource contentions (i.e. different machines simultaneously vying for the same disk drive, etc.).

After a brief description of the problem which was solved, I will explain how these drawbacks, or "bottlenecks", were either minimized or eliminated, producing very encouraging results.

The chosen problem comes from the field of Nuclear Engineering, while the method of solution (the Monte Carlo Method) comes from the field of Mathematics and has numerous applications. The technique of the Monte Carlo method is simple, yet elegant, using random numbers to solve several major types of problems including differential equations, integrals, and discrete gaming problems such as the one described later in this paper. Through this method, you can solve complex problems which would appear exhausting or even impossible through normal analytical methods. Figure 2 is a simplified sketch of one such problem. On the left is a source of neutrons, a nuclear reactor in this case. On the right is a reactor operator busy monitoring the system. The operator is shielded from the neutrons emitted in the reactor by a concrete wall of thickness "X". The Monte Carlo method is used to ensure that this wall is of adequate thickness assuring that the operations staff receives less radiation than the acceptable legal limits.

In the process of "diffusing" through the concrete wall, the neutrons undergo a number of different types of interactions, complicating their trip. One thing which might happen to the neutron is that it could make it through the concrete wall. This occurrence is shown in Figure 3. In travelling through the wall, the neutron moves in a straight line until it collides with an atomic nucleus. The smaller and much lighter neutron elastically bounces off the larger atomic nucleus at some random angle and

proceeds in a straight line until it collides again. We can simulate this process by using the APPLE's random number generator to choose a random angle. The distance the neutron travels before its next collision is also a random value. By keeping track of the neutron's position in the wall at any one time, we can track its progress which takes the shape of a zig-zag path.

Other things might happen to the neutron which would prevent it from reaching the control room area. As illustrated in Figure 4, the neutron may go through a series of collisions and ultimately be reflected back into the containment room. A third possibility is that the diffusing neutron may simply be absorbed by an atom within the concrete wall (see Figure 5). With all of these interactions going on, the problem complexity begins to grow as shown in Figure 6. The chance that these types of reactions will occur is statistical in nature, that is, there is a certain probability that each particular type of interaction will occur. These probabilities are determined by the characteristics of the particular materials used in the construction of the wall, the neutron's energy level, and various other factors. While this example is quite simplified, it is easy to envision a more realistic scenario where there might be several neutron sources (reactor, filters, liquid in piping, etc.), a number of walls or barriers, and numerous geometric problems (i.e. doors, ducting, corners, etc.). Through the use of the Monte Carlo method and the power of the computer, we are able to create a relatively simple program which is capable of tracking each neutron as it progresses, ultimately providing us with an otherwise impossible-to-achieve answer. We are able to effectively utilize microcomputers because the Monte Carlo method is simple and requires very little mass storage for calculations.

The first bottleneck previously mentioned, nonuniform parallelism (all processors not kept busy at the same time), is of no concern in our

situation because of the inherent parallelism of the Monte Carlo method of solution. The Monte Carlo method is a purely discrete process which is additive, that is, the more neutron "histories" followed, the greater accuracy achieved. Each microcomputer in the network simply adds its calculations to the overall results (which are stored in a text file on a floppy disk) in a timing sequence which minimizes disk access contentions.

The second bottleneck, that of contention over shared main memory, is bypassed by using a text file (i.e. external mass storage) and timing scheme to store the intermediate results. Additionally, the AROS operating system includes a program called SPEED which acts as a buffer. Briefly, the last 96 bytes accessed from the floppy disk are stored in the random access memory (RAM) of the Master (control) APPLE and are available almost instantly by any other user on the network. For example, if one user does a "CATALOG" of the floppy disk, this information is stored in the Master APPLE's RAM. If another user subsequently requests a "CATALOG," the information is immediately available on the screen, the system need not access the disk drive. Proper use of the SPEED program nearly eliminates disk accesses, thereby minimizing disk drive access contentions. Those of you who have suffered through the decision making process of a committee of more than a few people will have an appreciation for the importance minimizing contentions. If such a program were not available, the overall utility of the described network would be greatly reduced. In fact, if there were any but a very small number of microcomputers on the network, the system would be all but useless in practicality because the system users would spend most of their time waiting to access the disk drive(s). Typically, the use of the SPEED program with only a few machines on the network allowed over twice as many histories to be followed in a given time frame.

Proper "timing" of the code significantly affected results. A single machine running this particular Monte Carlo calculation was capable of tracking roughly sixty (60) histories per minute (1 history/second). When all eighteen (18) machines were used, this value rose to as high as nineteen hundred twenty (1920) histories per minute (32 histories/second). As the number of machines added to the network increased, it became important to alter the number of histories tracked before the calculations were added to the intermediate results in the text file. Fine tuning of this timing improved the results significantly. For a given amount of time, the number of histories tracked can vary by as much as a factor of one hundred (100) with eighteen machines on the network. These increases in speed are quite significant if you consider that a single APPLE took over five hours to follow 20,000 histories (a reasonable number for the accuracy I required) while the network was able to calculate the same 20,000 histories in only ten minutes.

In conclusion, there are specific groups of problems (systems simulation, discrete additive processes, etc.) which lend themselves to solution via multiprocessing. The economic advantages are appealing and become more so if the machines are utilized for other purposes when not employed as described. The described application takes advantage of network system capability which might otherwise go untapped. The use of systems such as this in the university environment further frees the mainframe as well as provides the individual researcher with an alternative to using only larger machines which are normally crowded throughout the school year.

Although low cost hardware (LSI & VLSI technology) is here today, I do not feel that the implementation of such networks will occur as rapidly as one might wish for the following reasons:

(1) The lack of viable, efficient multiprocessor (or network)

operating systems for the more popular microcomputers,

(2) Complexity of interprocessor communications,

(3) Configuration difficulties,

(4) Lack of proper development and evaluation software tools.

Correcting the first point, the lack of viable operating systems, will be the key to economic use, that is, the key to making these networks "tools" rather than "toys".

I wish to thank the following Professors from The Pennsylvania State University for their assistance: Dr. Anthony Foderaro (Nuclear Engineering) for the Monte Carlo program which I modified to run on the network, Dr.Edward Kenney (Nuclear Engineering) for sparking my interest in multiprocessing, and Dr. Bill Parks (Mechanical Engineering) for use of the AROS network.

Selected References

(1) Millikan, Roger C., "The Magic of the Monte Carlo Method", BYTE , February 1983, pp. 371-373.

(2) "Special Issues on Parallel Processors and Processing", Association for Computing Machinery Computer Surveys, Vol. 9, March 1977.

(3) Fuller, S. H., et. al., "Multi-Microprocessors: An Overview and Working Example", Proceedings of IEEE, Vol. 66, No. 2, 1978, pp. 216-228.

(4) Reyling, G., "Performance and Control of Multiple Microprocessor Systems", Computer Design , March 1974.

(5) Baer, J. L., "A Survey of Some Theoretical Aspects of Multiprocessing", Computing Surveys, Vol. 5, No. 1, March 1973.

(6) Korn, G. A., "Back to Parallel Co[illegible]tation", Simulation , August 1972, pp. 37-45.

(7) Street, R. T., "Micro-Multiprocessing", Mini-Micro Systems , May 1979, pp. 135-137.

(8) Hobbs, L. L., Parallel Processor Systems, Technologies, and Applications , Spartan, 1970.

(9) McGill, R. and Steinhoff, J., "A Multimicroprocessor Approach to Numerical Analysis: An Application to Gaming Problems", 3rd Annual Symposium on Computer Architechture, 1976, pp. 46-51.

(10) "Hybrid Monte Carlo Techniques with a Minimal Interface", Simulation , Vol. 17, December 1971.

(11) Dubois, M. and Briggs, F. A., "Performance of Synchronized Iterative Processes in Multiprocessor Systems", IEEE Transactions Software Engineering, SE-8, July 1982, pp. 419-431.

(12) Yue, W. Y. and Halverson, R. P. , "Making the Most of Multiprocessing for Microcomputers", Computer Design , February 1982, pp.101-106.

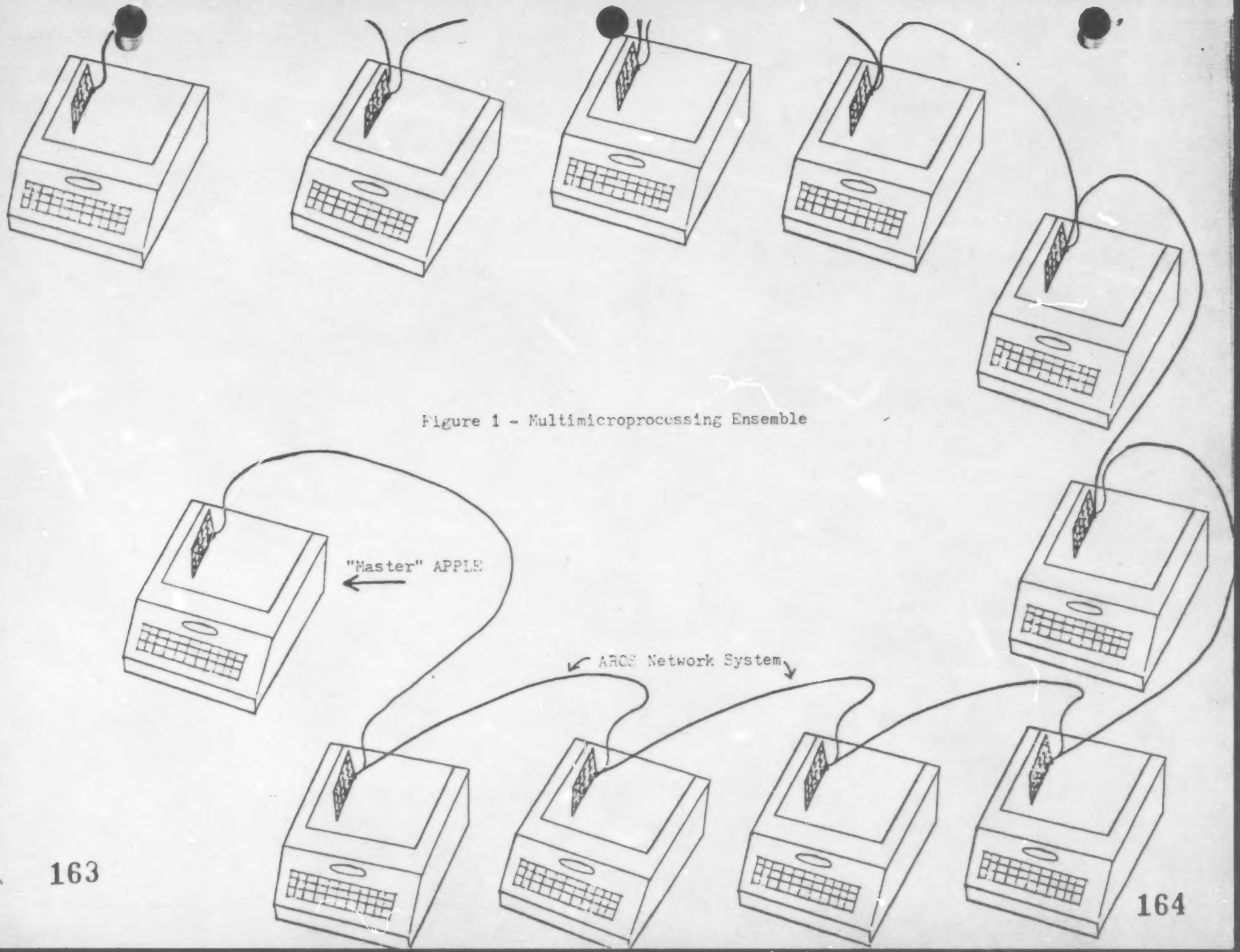


Figure 1 - Multimicroprocessing Ensemble

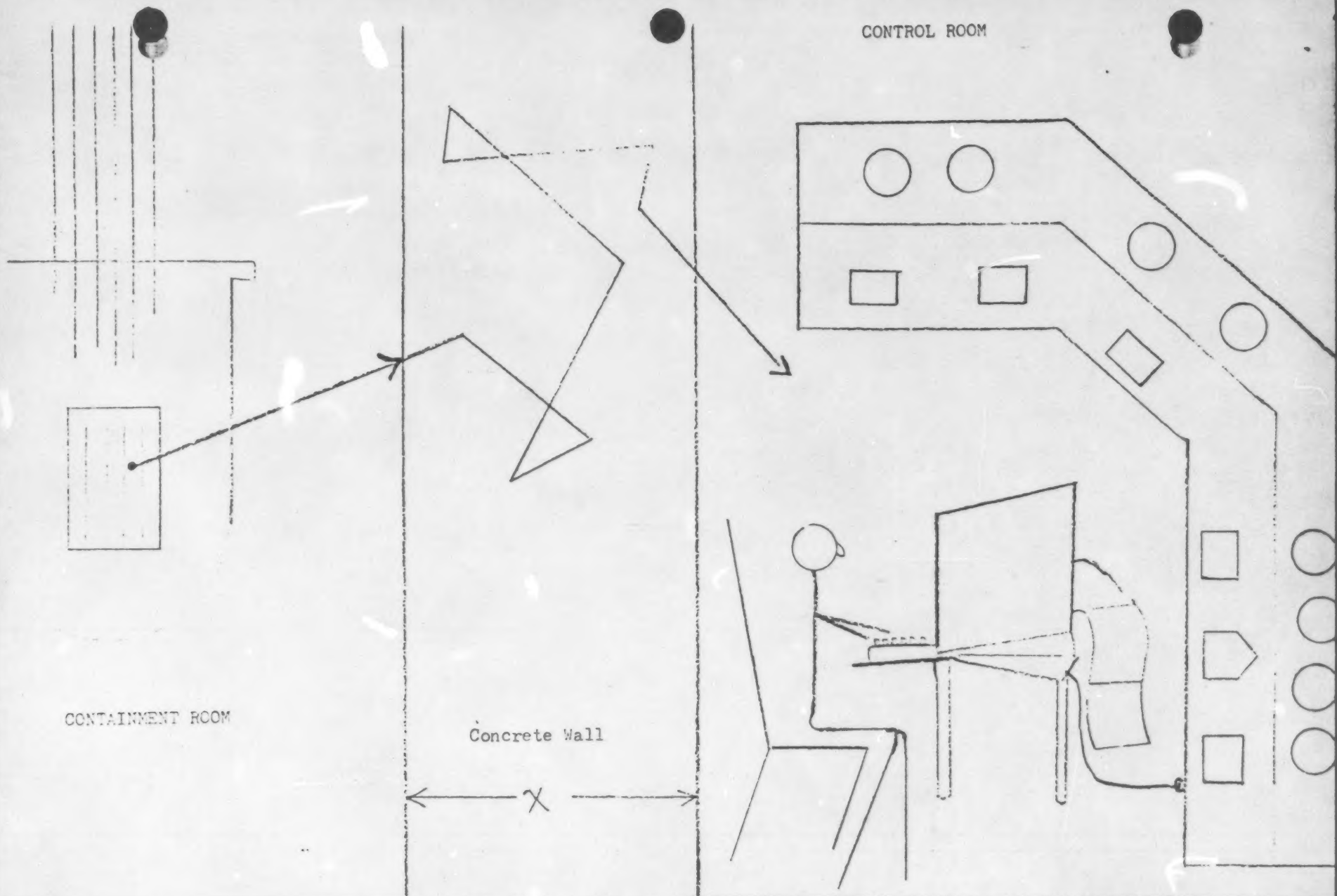


Figure 2 - Problem Sketch

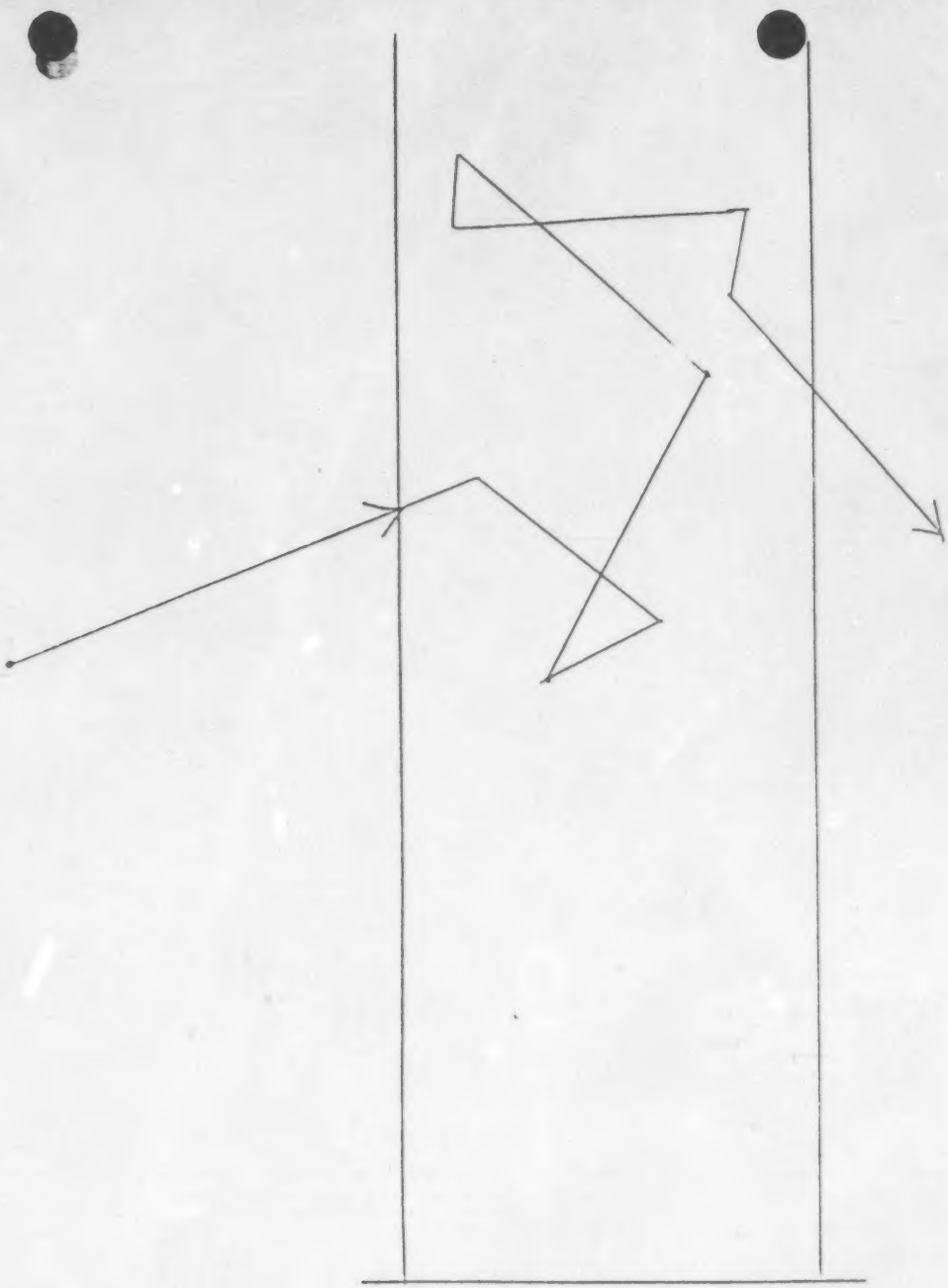


Figure 3 - Transmital

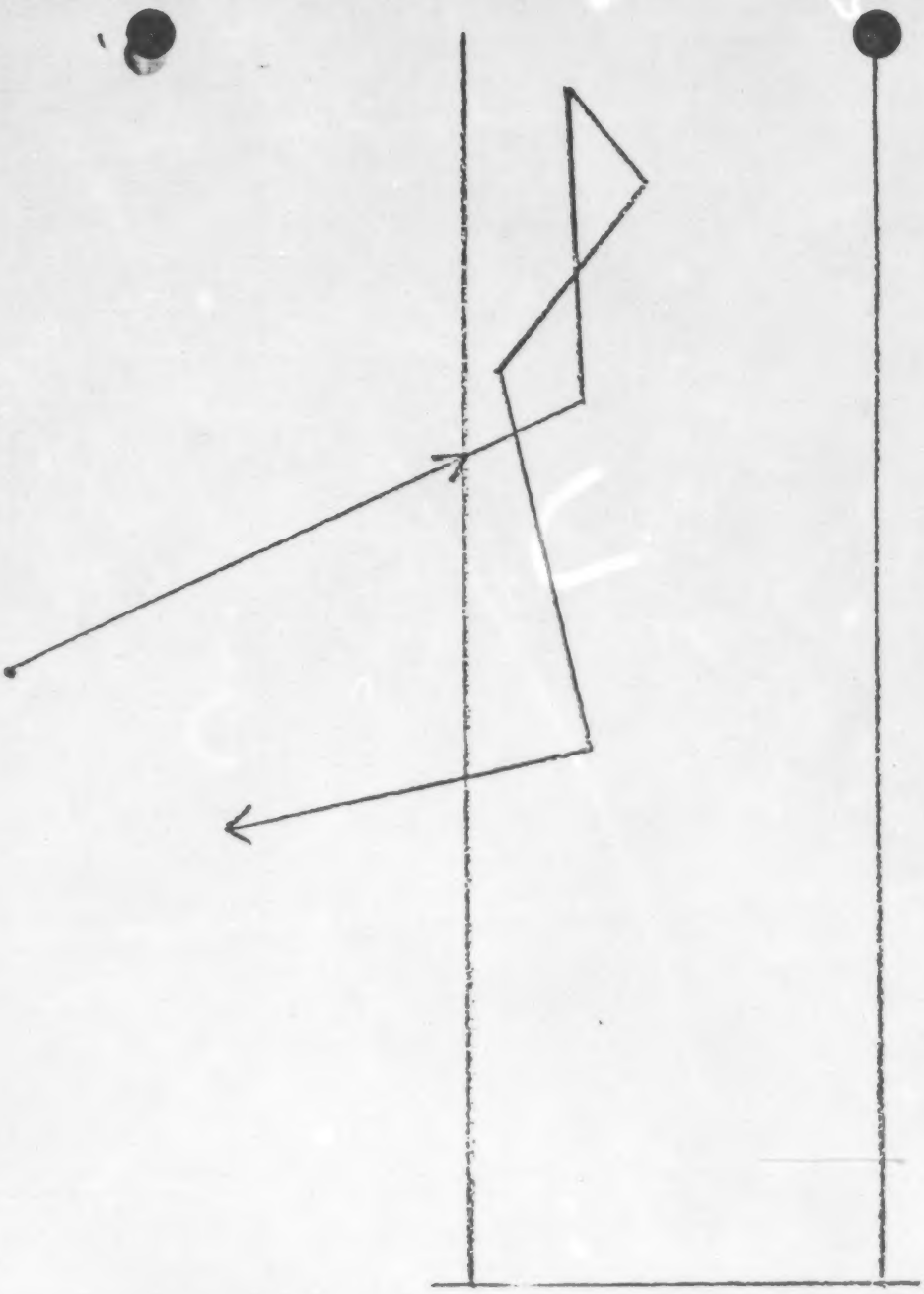


Figure 4 - Reflection

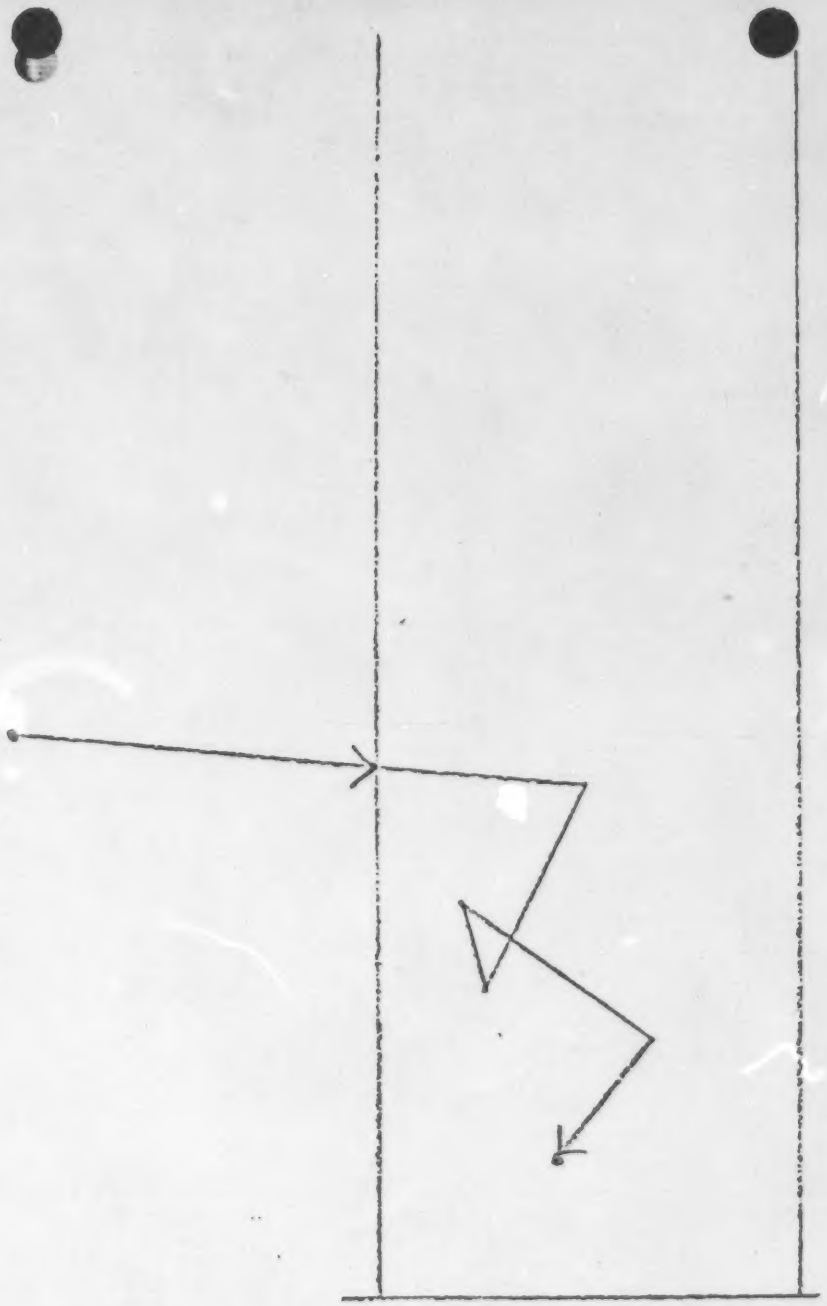


Figure 5 - Absorption

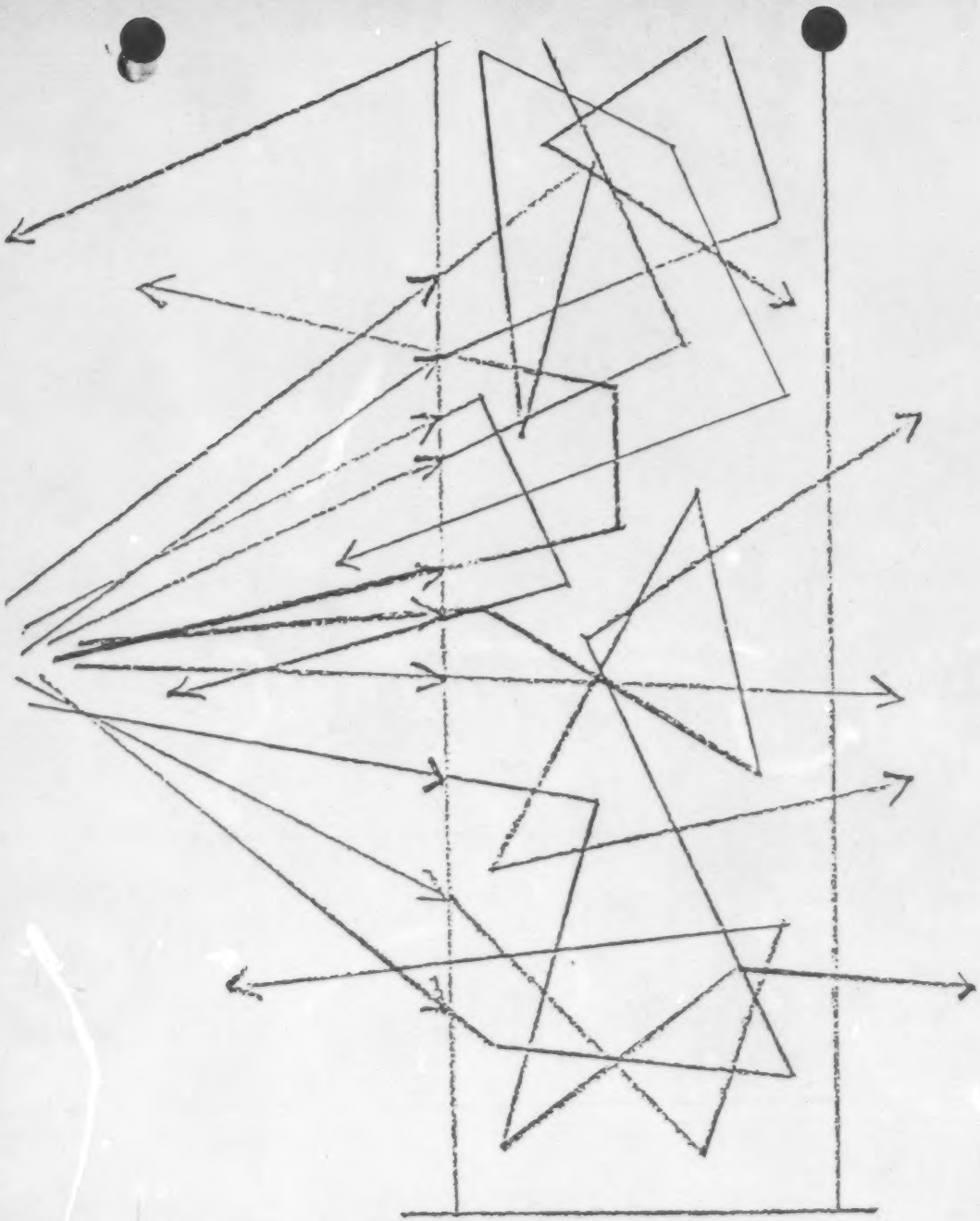


Figure 6

Microcomputer Applications for Laboratory Sciences

Dr. Thomas Sandusky
Watchung Hills Regional High School
and
Jeffrey D. Charney
Warren, New Jersey

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

Microcomputer Applications for Laboratory Sciences

Dr. Thomas Sandusky — Watchung Hills Regional High School
Jeffrey D. Charney — Warren, New Jersey

The computer has become an integral part of our life. This fact was made dramatic by Time nominating the computer as its 1982 "Machine of the Year". This distinction has always gone to noted world figures.

> There are some occasions, though, when the most significant force in a year's news is not a single individual but a process, and a widespread recognition by a whole society that this process is changing the course of all other processes. That is why, after weighing the ebb and flow of events around the world, TIME has decided that 1982 is the year of the computer. (Friedrich, 1983:16)

At first, the computer s impact was very distant. It was this large, onerous machine that sent bills or kept track of our debt to the government. It was a mysterious machine and usually a negative experience when we had to interface with it.

Not too long ago the computer started to take on a new image. We didn't know it but this little ball being bounced back and forth on our television was the beginning, the fringe of an invasion. Figures from outer space descended upon us. We fought them from places of recreation and then our homes. The computer entered our homes in disguise. It was fun. How were we to know the video-game was the computer's foot in the door?

In 1975, the microcomputer was born and our society will evolve with the technology of the microchip. Playing the video-games on the microcomputer will be the primary motivation for purchasing a computer for most families. But as time goes on the computer will be a necessity.

The computer will make our homes energy efficient by monitoring heating in the winter and air conditioning in the summer. It will be our "watch-dog" by opening drains, turning lights on and off and telling us when an intruder is breeching the sanctity of our home. The computer will be our servant by waking us in the morning, starting our coffee and breakfast, planning and preparing meals that match our

dietary needs. We may find little robots scurring around keeping the place in order. Our friend will manage the family's finances. The bills will be paid; budgets planned; investments monitored and taxes prepared without touching a dollar or a checkbook.

Chances are that our future employment will deal with information processing and communications. Already, the largest single class of employees in the United States have jobs in the computer and communications industries (Porat, 1977). This will continue to grow until every job and profession will either use or service the computer and communications technologies.

As Educators, we have to integrate our students into this high technology society. Arthur S. Melmed (1982) indicates that schools have begun to acquire computer technology on a modest scale. This includes courses on computer literacy, drill and practice for improving basic skills, teaching programming in advanced math courses and simulation in the sciences. A modest amount of hardware exists but the software is, for the most part, not viable in the classroom.

As science educators, we are concerned with how to integrate the computer into science instruction. The research indicates that teaching old curricula on the computer does not have an advantage over traditional instructional forms (Edwards, et.al., 1975; Jamison, et.al., 1974; Kulik, et.al., 1979). The commercial software is basically unimaginative and does not improve upon contemporary teaching methods (Euchner, 1983). Much of the existing software presents approaches to learning that differ little from a "fancy textbook". Other areas such as drill and practice on the computer, and graphing exercises have been successful in the area of basic math skills but in science courses we want to go beyond lower levels of the learning heirarchy.

The heart of the science course is the laboratory. Here is a setting where the student designs experimental procedures to test hypotheses, collects data,

makes inferences based on the date, makes observations, etc. The laboratory is an independent, participatory activity that provides a vehicle for higher learning such as analysis, problem solving and evaluation. This is an ideal situation for students to develop their higher mental abilities. But, it has been experienced that in actuality, only about twenty to thirty percent of the class mentally apply themselves. The remainder of the class "tag-along", copying thoughts and conclusions of the students who practice analysis, problem solving and evaluation. A teacher has difficulty proving which students came up with the original answers on the laboratory write-ups that are handed in. There is limited time to develop and grade an evaluation instrument that will discriminate which students applied themselves during the lab. This is an instructional problem the computer can solve and an excellent method for introducing the science student to the computer.

Software developed to engage science students in alalysis, problem solving and evaluation should have the following characteristics: it should relate to the curriculum; allow students to display data in graphic form; engage students in "higher-order skills"; provide reinforcement and diagnostic help; permit students to control the speed of the program, and use a presentation format that is consistent and easy to read.

Software should supplement the objectives of a unit of instruction. Traditional teaching methods such as lecture, textbooks, films etc. should introduce the subject materials. Large group instruction is just as effective as the computer for presentation of content and until the computer is used in the home as an extension of the school, traditional methods are more cost-effective.

Such an example of large group instruction is the laboratory. A laboratory is designed to apply concepts and rules learned during large group instruction. Data is collected during the laboratory. The computer then requires the student to input and manipulate collected data in order to predict or generalize information

to other laboratory or "real-life" situations. The program also should involve the learner in problem solving. Hypotheses can be supported or not supported based on how the learner interacts with the data. The program can also ask the learner to explore why hypotheses were not supported.

Feedback should be used to reinforce correct responses and provide diagnostic help for non-correct responses. The diagnostic help can be built into the program or request the learner to see the teacher for help.

Another vital aspect of computer instruction is the time element. Students should be able to control the pace of the program. Individuals differ in the amount of the time to read, think or calculate.

The presentation of information should be consistent throughout the program. Avoid overcrowding the amount of text on the screen. Introduce tables or graphs. Do not display the data without explaining what is going to be presented. Poor presentation of text can intefere with the learning process.

Software should evaluate the learner s performance during the program. Evaluation techniques will hold the individual learner accountable for mastering unit objectives. Students will not be able to "tag-along" during lecture and laboratory. They will have to get involved in the learning process in order to achieve at a high level.

We are in the process of developing biology software that is designed around the principles described. The initial program was a supplement to a cell lab. The second program supplemented a bacteria respiration/fermentation lab. Encouraged by the success of the first program, students were asked to evaluate the respiration/fermentation program. The students involved included eighty-eight ninth grade biology students. The four classes used were in an advanced tract of college-bound students.

The students were required to input population data on bacteria grown with air and without air. Different concentrations of food (glucose) were also used in each

group of bacteria. Analysis questions were asked based on the data. The students were asked to contrast the two sets of data and calculate the differences. They were also required to compare the two sets of data in order to establish how many times greater the growth of bacteria with air was to bacteria grown without air. The students were required to predict growth beyond data collected. These types and other related questions verified that students were achieving at a comprehension level.

If concepts of respiration and fermentation were learned during presentation and lab instruction, students could explain the following:

> Some bacteria are able to survive thousands of years by producing a capsule to protect themselves. Prehistoric mammals such as wooly mammoths and sloths have been found in glaciers and bogs, but decay rapidly when exposed to air. What explains this phenomenon?

A scenario was stated and students were expected to solve the following problem based on their knowledge of respiration and fermentation.

> You accidentally drop the flask containing this deadly bacteria and run for your life. You smartly seal the lab door. What should you do next to save the human race?

These questions were evaluated as involving comprehension, analysis and problem solving skills.

The evaluation instrument used for this study revealed that ninety-four percent of the students had a positive attitude about working through the program, but more than half (56%) of the students were dissatisfied with their grades.

The presentation and format of the program was examined to explain the poor performance. Except for a few statements, the instructions were clear and easy to understand. The majority of the students understood what the graphics represented; only a few grammatical errors were detected and the program went from one part to another without difficulty. Poor presentation was not the reason for more than fifty

percent of the students achieving lower than usual on a laboratory activity.

Eighty-eight percent of the students indicated that they were looking forward to further work on bacteria respiration/fermentation. Ninety-one percent of the students stated that the subject matter of the program related to the unit of respiration. These two items are designed to measure the student's motivation to learn. Bloom (1976) defines this as subject-related affect which is measured by asking the student if he/she would like to study the subject in the future. The subject-related affect is also a measure of the student's perception of the relation between the present task and some larger set of goals, purposes, or objectives. This would establish whether the material is relevant to the student. According to Bloom (1976) the subject-related affect can account for as much as twenty percent of the variance on a measure of achievement. The attitude or affect of the students toward the subject matter of the program was positive. The subject-related affect does not account for more than fifty percent of the students achieving lower grades.

The program provided feedback. Reinforcement techniques were used for correct responses and the program gave diagnostic help for incorrect responses. Sixty-one percent of the students indicated that the feedback was very helpful. Thirty-eight percent responded that they appreciated the feedback but it was not vital for high achievement. Only one percent of the students felt that the feedback was a waste of time.

The students' attitude toward the program was positive. Technically, the program did not interfere with learning and the feedback was generally helpful to the students. After ruling out these variables, the poor performance of more than one-half of the students can be directed back to the problem stated earlier. In actuality, only twenty to thirty percent of the students mentally apply themselves to a laboratory activity. The "tag-alongs" did not invest their time to develop

their higher mental abilities. This was well-stated on the evaluation by a student:

"It was easy if you prepared beforehand."

It is proposed that if the computer is used to engage the individual student in analysis, problem solving and evaluation activities based upon data collected during a laboratory activity and evaluate their performance, a much larger percent of students will develop higher mental abilities. Herein lies one of the computer's greatest assets: its technological ability to direct and evaluate the learning process much faster than a teacher. The students also would not only have to memorize content but they must be able to comprehend the subject matter and be able to apply it during a laboratory activity. Then the computer can be an effective instructional tool when students use information generated in presentation and laboratory activity. Software should be written which provides an interactive relationship with learners. It should allow the learner to branch to various activities based upon the learner's interpretation of data. This is the ideal match between technology and learning.

REFERENCES

Bloom, Benjamin S. "Human Characteristics and School Learning". New York: McGraw-Hill, 1976.

Edwards, J., and others. "How Effective Is CAI? A Review of the Research". Educational Leadership 33 (November, 1975): 147-153.

Euchner, C. "States Begin Coordinating School Computer Use". Education Week, Volume 2, Number 21 (February 16, 1983): 15.

Friedrich, Otto. "The Computer Moves In". Time (January 3, 1983): 14-24.

Jamison, D.; Suppes, P.; and Wells, S. "The Effectiveness of Alternative Instructional Media". Review of Educational Research 44 (1974): 1-67.

Kulik, J.; Kulik, C.J. and Cohen, P. "Effectiveness of Computer-Based College Teaching". Unpublished Paper, University of Michigan (1979).

Porat, M. RHW Information Economy. Department of Commerce, Office of Telecommunications and The National Science Foundation, U.S. Government Printing Office, Washington, D.C.: 1977.

Melmed, Authur S. "Information Technology for U.S. Schools." Phi Delta Kappa (January 1982) 308-311.

Computer-Managed-Instruction:
Will I Control It or Will It Control Me?

Ms. Carol Scheftic
Graduate Student
University of Pittsburgh

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

"Alphabet Soup"
COMPUTER-MANAGED INSTRUCTION

1. Systems

- AIMS - Automated Instructional Management System

  * New York Institute of Technology
    part of GEMS - Generalized Educational Management System
    Alexander Schure
  * The Pennsylvania State University
    Center for Cooperative Research with Schools
    John Hayman, Cynthia Mable

- AIS - Advanced Instructional System
  developed by McDonnell Douglas Corp.
  with the Air Force Human Resources Laboratory
  AFHRL-TR-79-40 December, 1979
- Patrick Suppes

  * CCC - Computer Curriculum Corp.
    1070 Arastradero Road
    Palo Alto, Ca 94304
    (800) 982-5851 (Calif)
    (800) 227-9324 (other)
  * Institute for Mathematical Studies in the Social Sciences
    Ventura Hall
    Stanford University
    Stanford, CA 94305
    (415) 497-3113

- CIMS - Cincinnati Instructional Management System
- CISS - Computerized Instructional Support System
  New York Institute of Technology
- COMPUTRAC
  Florida State University
  John Healy
- CTSS - Classroom Teacher Support System
  IBM
  Gerald Lippey and others
- COMIT
  Waterloo, Ontario, Canada
  Symposium on COMIT - April, 1978
- IMS - Instructional Management Systems
  System Development Corp.
  John E. Coulson, Cleone Geddes, Beverly Kooi
- LRDC - Learning Research & Development Center
  University of Pittsburgh

Pittsburgh, PA 15260
CMI for IPI (Individually Prescribed Instruction)
- MICA - Managed Instruction with Computer Assistance
Wisconsin Laboratory of Experimental Design
- NDPCAL - National Development Programme in Computer Assisted Learning

  * Nicholas J. Rushby
    Ingersoll College
  * John Fielding
    Council for Educational Technology
    3 Devonshire Street
    London W1N 2BA England

- PLAN - Program for Learning in Accordance with Needs
Westinghouse Learning Corp.
Iowa City, IA
- PLATO - Programmed Logic for Automated Teaching Operations
developed at University of Illinois
Computer Based Education Research Laboratory
marketed by Control Data Corp.
- SCOUT
Brigham Young University
- SPIMS - Student Performance Information Management System
University of Utah
Robert Steffenson, Edwin Read
- TICCIT - Time Shared Interactive Computer Controlled Information Television
developed at Brigham Young University
by Mitre Corp.
with NSF aid
marketed by Hazeltine Corp.
- TIPS - Teaching Information Processing System
University of Wisconsin
Allen Kelly
- USN-CMI - U. S. Navy
Navy Personnel Research and Development Center

  * San Diego,CA
    John Ellis, Nick van Matre
  * McLean, VA
    Kenneth Polcyn
    COMSAT

- WIS-SIM - Wisconsin System for Instructional Management

2. Interested People & Organizations

- Robert Albrecht
People's Computer Company

1263 El Camino Real
Menlo Park, CA
(415)323-3111

- Karen Billings
  Director, Microcomputer Resource Center
  Teacher's College, Columbia University
  New York, NY 10027
  (212)678-3740
  Co-Author of "Are You Computer Literate?"
- William Bozeman
  University of Iowa
- Alfred Bork
  Director, Educational Technology Center
  University of California at Irvine
  Irvine, CA
  (714) 634-5011
  from CAI physics to
  a new model of courseware development
- Ludwig Braun
  Laboratory for Personal Computers in Education
  SUNY - Stony Brook
  Stony Brook, NY 11794
  (516) 246-8424
  Huntington Project - simulations
  microcomputers in schools
- Sylvia Charp
  Director of Instructional Systems
  The School District of Philadelphia
  Fifth and Luzern Streets
  Philadelphia, PA 19140
  (215) 229-9492
  as of 1979, Phila. had 50,000 students
  in 30 schools using 240 terminals and 55 micros.
- CONDUIT
  100 LCM Building
  University of Iowa
  Iowa City, IA 52242
  (319) 353-2121
  Clearinghouse for college CAI courseware
- Christopher Dede
  Chariman, Education Section
  World Future Society
- Tom Dwyer
  Project SOLO
  Computer Science Dept., Alumni Hall
  University of Pittsburgh
  Pittsburgh, PA 15260
  (412) 624-5357
- MicroSIFT
  Judith B. Edwards, Director
  Computer Technology Program
  Northwest Regional Educational Laboratory

Portland, OR 97200
(503) 248-5900
Clearinghouse of K-12 micro materials for teachers

- Raymond Fox
  Society for Applied Learning Technology
- Jacques Hebenstreit
  Chairman, Computer Science
  Ecole Superieure d'Electricite
  10 Avenue Pierre La Rousse
  92 Malakoff, FRANCE

  * 1970-75: standard hardware configuration and software language defined, created, and mandated for use throughout secondary schools.
  * 1979-85: plan to place 10,000 microcomputers in French schools for children aged 11-18.

- Illinois Series on Educational Application of Computers
  School of Education
  University of Illinios
  Urbana, IL
- Arthur Luehrmann
  Director of Computer Research
  Lawrence Hall of Science
  University of California - Berkeley
  Berkeley, CA 94720
  (415) 642-4193
  using computers in museum setting
  providing access to computers in schools
- Nicholas Negroponte
  Prof. of Computer Graphics
  Architecture Machine Group
  Massachusetts Institute of Technology
  Cambridge, MA 02139
  (617)253-1000
  "Media Room" concept
  much of his support comes from the Office of Naval Research and the Defense Advanced Research Project Agency
- MECC - Minnesota Educational Computing Corp.
  2520 Broadway Drive
  St. Paul, MN 55113
  (612) 376-1122
  John Haugo - was Director until 1980, many publications under his name
- Seymour Papert
  Project LOGO
  Artificial Intelligence Laboratory
  Massachusetts Institute of Technology
  545 Technology Square
  Cambridge, MA 02139
  (617) 253-6214

Children program/control a "Turtle"
Child initiates the action, not the computer.
See also Papert's book Mindstorms: Computers, Children, and Powerful Ideas, Basic Books

- Howard A. Peelle
  Director, Instructional Applications of Computers
  School of Education
  University of Massachusetts
  Amherst, MA 01003
  (413) 545-0496
- WICAT
  University Plaza, Suite 201
  1160 South State Street
  Orem, UT 84057
  (801)224-5400
  Dustin H. Heuston, Chairman
  C. Victor Bunderson, President (TICCIT designer)
  a nonprofit institute to help education through technology
- Xerox Palo Alto Research Center
  Learning Research Group
  3333 Coyote Hill Road
  Palo Alto, CA 94304
  (415) 494-4000
  Allen Kay, Director
  John Seely-Brown - application of AI to instr. comp.
  Ira Goldstein - computer as intelligent coach, has worked with Papert
- Karl Zinn
  Center for Research on Learning and Teaching
  University of Michigan

Computer Courseware for the Deaf and Learning Disabled: What's Being Done

Ms. Casey Stone and Ms. Kathleen Smith
Instructional Systems Graduate Students
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

Computer Courseware for the Deaf and Learning Disabled: What's Being Done?

Casey Stone

Kathleen Smith

In our technological society, special education has historically demonstrated an openness to innovation by taking advantage of many technical devices to maximize the development of handicapped individuals and to help overcome disabilities. However, innovations in the area of computer technology are not equally represented among the various handicapping conditions, particularly with regard to communication disorders. The Congressional Subcommittee on Domestic and International Scientific Planning, Analysis, and Cooperation and The Subcommittee on Select Education held hearings in 1977 specifically concerned with advances in computer technology for the handicapped. Although the conference ostensibly included all the handicapped, the technological devices that were shared dealt exclusively with the blind and motorically handicapped. Again in 1978, at the 5th Annual Conference on Systems and Devices for the Disabled, there was no mention of technology for the cognitively or perceptually impaired or learning disabled person, and only two of the papers presented dealt with deafness. The West Coast Computer Faire in 1978 reflected this same bias (Goldenberg, 1979). By October of 1981, the situation had improved slightly. Of the 97 papers published in the Proceedings of the Johns Hopkins First National Search for Applications of Personal Computing to Aid the Handicapped, 11 dealt with educational applications of computers for the learning disabled and deaf. Although the first applications of computer technology is gaining acceptance in hearing impaired programs, most of the applications can not be considered true innovations because they use teacher made adaptations of extant courseware. Because of the technology's general neglect of the needs of the learning disabled and deaf,

and because of their similar learning deficiencies, we have chosen to focus on these two populations.

For the purpose of this paper, learning disabled children are those "who have disorders in one or more of the basic psychological processes involved in understanding or in using spoken or written language. This disorder may manifest itself in imperfect ability to listen, think, speak, read, write, spell, or do mathematical calculations..." (Harris, 1980). Deaf children are those whose hearing loss is such that they are unable to understand and interpret the sounds within the range of normal speech. Language reception is therefore severely inhibited. "The primary handicap imposed by prelingual deafness is a retardation and distortion of the deaf child's language development" (Galbraith, el al., 1979). Based on generally accepted prevalence rates, there are approximately one-half million school age deaf children in the United States and another half million children with specific learning disabilities (Goldenberg, 1979). Unfortunately, based on a review of the literature, a minimum amount of computer courseware has been developed to meet the specific needs of these populations; likewise, design guidelines have never been defined (Richardson, 1982). Overall, there has been a lack of any cohesive effort in this area.

Despite these conditions, a recent survey of microcomputer usage in schools for the deaf, conducted by the Educational Media Production Project for the Hearing Impaired, found that 42% of the responding schools are currently using microcomputers. Of the 158 schools surveyed, 120 responded, representing 45 states. Fifty schools indicated they have microcomputers. When asked

whether they were considering the purchase or rental of additional microcomputers, 67 programs said yes, 41 said no, and 12 made no response. The major use of the computers was for student instruction and use, taking 62% of the computer time, while the teaching of computer languages and programming occupied the computers another 16% of the time. When asked to cite the major problems with using microcomputers, the schools' most common response was the lack of suitable software. Specific problems that were mentioned relating to available software included difficulty of the language and vocabulary, utilization of audio tones, understanding and following directions, too much material on the screen at one time, and print size being too small (Harding and Tidball, 1982). Although no comparable surveys were found for computer usage in programs for the learning disabled, research indicates that common problems exist. Schools are investing in computer systems for classroom usage, but are having a difficult time finding appropriate courseware.

Some of the more effective courseware that is being utilized is in the area of exploratory learning. An observer in a program for learning disabled students which used this approach, noted the element of play involved in learning, with students actively involved. The computer had the capactiy to become a partner for anyone who wanted to use it. As one child stated, "This is where the action is!" (Frillman, 1982). One of the simplest devices that has been used with kindergarteners is the programmable robot, BIG TRACK. Without the child needing to know how to read or write, the robot can be instructed by the child to go forward, backward, right, and left for a specific distance. By estimating the proper

sequence of instructions to move BIG TRACK through a maze, children extend their numerical concepts, directionality, logical thinking, and self concept (Shanahan, 1982). This device is very similar to the LOGO turtle. LOGO, which is a general-purpose computer language, allows children to teach the computer to perform various feats without depending on a particular curriculum. Instead of having a smart computer teach a "dumb student," LOGO encourages students to think of themselves as intelligent while they learn to teach "dumb computers." LOGO has been successful with deaf and learning disabled students in developing skills in reading, spelling, typing, math and spatial reasoning. In addition, it has helped to develop logical thinking and problem solving skills, as well as fostering independence of action, increased self-esteem, and greater self-knowledge. Through the use of LOGO, learning disabled and deaf students have made significant growth in both skills and motivation (Weir, 1983).

A system designed specifically for children with communication handicaps is CARIS (Computer Animated Reading Instruction System) (Geoffrion and Goldenberg, 1981). Assuming the student has no initial reading vocabulary, CARIS teaches reading through the exploration of the meaning of noun-verb sentences. From lists of nouns and verbs, the student, using a light pen, selects two words to create a noun-verb sentence which the computer then illustrates through animated cartoons. As reading and writing skills improve, more adanced versions require students to actually type in the words. Another language oriented system being designed for this population is Iliad (Interactive Language Assistance for the Deaf), which is currently under development at Boston

University. The program operates to a large extent under student control and provides tutorials in both the production and comprehension of written English. Iliad should be available in another year (Le Cavalier et al, 1982).

Why do these and similar systems appear to be effective with deaf and learning disabled populations? Although there is little empirical data to support any conclusions, the strengths of the systems seem to lie with their exploratory nature and minimal dependence on reading ability. Students are able to progress at their own rate and along their own paths, using errors as learning experiences in a highly interactive environment. Because the programs incorporate high visual components rather than depending on reading skills, students are able to capitalize upon their strengths rather than their weaknesses. And although language development is not always the primary objective of the systems, improved communication skills are often an outcome. The programs help children use the language skills they already have and through communicative interaction, help to expand and improve these skills. Students use language to ask questions and control their environment, as well as to answer questions, in a manner similar to the typical interactions of young children. According to Goldenberg, "Communication is not the product of language development, but rather the opposite--language development results from communicating" (Geoffrion and Goldenberg, 1981).

Computers can also provide a programmed approach to learning which is an indispensible instructional strategy for deaf and learning disabled students (Le Cavalier, et al, 1982). Because of their information processing deficiencies, these students do

not experience the repetition and redundancy of information input that non-handicapped populations receive on a daily basis. Therefore, it is necessary to provide this repetition within a well structured framework where instruction proceeds in small increments and children progress at their own rates. Computer assisted instruction (CAI) is one way to provide this structure. Research has shown the "positive effects CAI support can have on academic achievement and on the motivation levels of deaf students (Richardson, 1981).

The Institute for Mathematical Studies in the Social Sciences (IMSSS) Project at Stanford University, under the direction of Patrick Suppes, was one CAI project perceived of as positive by both faculty and students at the California School for the Deaf at Berkeley/Fremont and at the Florida School for the Deaf and Blind. This project utilized drill and practice strategies in the areas of elementary math and language arts. Over 180 terminals were connected to the system from 1970-1973, after which time the project was terminated due to a loss of federal funding. Although student progress in language arts was not clearly attributable to the system, gains in computation skills were 2-3 times greater than what was expected from classroom instruction. The system demonstrated the capactiy to respond appropriately to the individual learning pace of students, as well as to provide individualized instruction at a reduced cost. When the IMSSS Project folded in 1973, the Florida School for the Deaf and Blind instituted a similar CAI project through Computer Curriculum Corporation. Each of the lessons in math, language, and reading was designed specifically for individual students so that areas

of strength were expanded and areas of weakness remediated. Early research indicated that the deaf students who normally progress at the rate of .5 years for each year of instruction, were improving in math at the rate of approximately one year during six months of instruction with this system. Gains were also made in the areas of reading and language (Hoffmeyer, 1980).

The PLATO system, which has been used in several schools, has also shown to be similarly effective. One representative project was conducted in the Carrie Busey Elementary School in Champaign, Illinois, where PLATO was a regular part of the math program for both handicapped and non-handicapped students. Using any combination of the 250 lessons in the PLATO data bank, teachers develop instructional modules for individual students. The teacher assigns the topic, sets the length of the sessions, and sets the parameters related to the student's previous experiences. Results indicate that the program has a strong positive effect on student achievement in mathematics. It has also helped children to assume a greater degree of responsibility. Knowing that they must complete one lesson before they can go on, students learn to make decisions about what to do next. Although the programs generally utilized a minimum of written language, they helped to improve students' communication skills. As a result of following directions and giving directions to the computer, students developed language of direction. Reading skills also improved and vocabularies grew as students were motivated to succeed. Finally, as students shared their work and experiences, they learned to communicate about their mathematics (Dugsdale and Vogel, 1978).

It is clear that drill and practice programs, as well as exploratory programs, can have a positive effect on learning with the deaf and learning disabled. The visual and graphic dimensions of the computer can advance the potential learning styles for language deficient students (Frillman, 1982). When properly used, the computer is a motivating teaching tool which increases student enthusiasm toward learning. Students feel a sense of responsibility and accomplishment. They can progress at their own rate of learning, while receiving both positive and negative feedback in non-threatening ways. Reading is reinforced through interaction and students show renewed interest in language areas. With privacy and without embarrassment, they have the opportunity "to see themselves succeed without the frustration that competition fosters" (Hoffmeyer, 1980). One example of this is in the area of word processing where students are motivated to write and revise because they can be proud of their final products. They no longer have to experience laborious erasing, rewriting, and personal discouragement (Shanahan, 1982). "The by-products of exposure to computer-based education may be as important a consideration as the courseware itself" (Richardson, 1981).

Although the potential of both exploratory and drill and practice CAI materials has been demonstrated for deaf and learning disabled students, many of the programs currently available are not appropriate for this population. Courseware is often highly language dependent, with extensive written instructions and other textual materials. Graphics, when used, tend to act as adjuncts to the written information rather than convey content. For deaf and learning disabled students, this environment is highly

frustrating because most information is being transmitted through their weakest modalities. Reaearch indicates that schools utilizing CAI courseware often find it necessary to rewrite the text so that it can be understood by the students (Castle, 1982; Garvey, 1982; Harding and Tidball, 1982; Le Cavalier, et al, 1982; and Slovick, 1982). Other revisions that are often required include adding differential levels to accommodate the heterogeneity of the population, breaking instruction into smaller segments, increasing the number of responses available, adding new branches and including action in order to keep students' attention (Castle, 1982; Frillman, 1982; and Harding and Tidball, 1982). Considering the numbers and extent of the revisions which often must be made, it is no wonder that the California School for the Deaf at Fremont has developed its own course authoring system.

Use of computers with the handicapped is still in the early phases and many of the reports contain little or no empirical data (Williams, et al, 1983). But based on preliminary studies, it is apparent that the potential exists and that computer usage will continue to grow and expand. From the work that has been done, it is clear that computers hold promise for meeting the individual needs of both deaf and learning disabled students, not only in teaching specific content but also in improving language skills. Informal evaluations have determined some of the components necessary for success, but most software development in this area is based on untested hypothesis. It is imperative that research be conducted to establish parameters and guidelines for designing effective courseware.

Arcanin, Jacob. Computer-assisted instruction at the California School for the Deaf- past, present, and future: an administrator's view. American Annals of the Deaf, 1979, 124 (5), p. 573-577.

Arcanin, Jacob & Zawolkow, Geoff. Microcomputers in the service of students and teachers -- computer-assisted instruction at the California School for the Deaf: an update. American Annals of the Deaf, 1980, 125 (6), p. 807-813.

Castle, Daniel J. Mathematics software, a computer lab, and the hearing impaired. The American Annals of the Deaf, 1982, 125 (5), p. 495-504.

Comden, Tippi. The many uses of Apple computers at the Western Pennsylvania School for the Deaf. American Annals of the Deaf, 1981, 126 (6), p. 591-599.

Dugdale, Sharon & Vogel, Patty. Computer-based instruction for hearing-impaired children in the classroom. American Annals of the Deaf, 1978, 123 (6), p. 730-743.

Durkee, David. Maple sugar and apples: a yummy way to learn. Softalk, 1982, 2 (4), p. 170-174.

Frillman, Louis W. The development of your own programs for the microcomputer -- a process approach. American Annals of the Deaf, 1982, 127 (5), p. 625-630.

Galbraith, Gary, Lennan, Robert, and Peterson, Barbara. Interfacing an inexpensive home computer to the videodisc: educational applications for the hearing impaired. American Annals of the Deaf, 1979, 124 (5), p. 536-541.

Garvey, Mary. CAI as a supplement in a mainstreamed hearing-impaired program. American Annals of the Deaf, 1982, 127 (5), p 613-616.

Geoffrion, Leo D. Reading and the non-vocal child. The Reading Teacher, 1982, 35 (6), p. 662-668.

Geoffrion, Leo D. and Goldenberg, E. Paul. Computer-based exploratory learning systems for communication-handicapped children. The Journal of Special Education, 1981, 15 (3), p. 325-332.

Goldenberg, E. Paul, Special Technology for Special Children, Cambridge, Mass.: University Park Press, Baltimore, 1979.

Harding, Richard E. and Tidball, L. Kaye. A national microcomputer-software survey of current microcomputer usage in schools for the hearing impaired. American Annals of the Deaf, 1982, 127 (5), p. 673-683.

Harris, Albert J., An overview of reading disabilities and learning disabilities in the U.S., The Reading Teacher, 1980, 33 (4), p. 420.

Hoffmeyer, Dennis B. Computer-aided instruction at the Florida School for the Deaf and Blind. American Annals of the Deaf, 1980, 125 (6), p. 834-840.

Joiner, L.M., Sedlak, R.A., Silverstein, B.J. and Vensel, G. Micro-computers: an available technology for special education. Journal of Special Education Technology, 3 (2), Winter 1980, p. 37-42.

Le Cavalier, Diane, Vaughn, Susan, and Wimp, Marcie. Putting microcomputer technology to work for our hearing-impaired students. American Annals of the Deaf, 1982, 127 (5), p. 512-521.

Microcomputers' Place in Special Education, Exceptional Children, Reston, Virginia: 49 (2), October, 1982.

Proceedings at the Johns Hopkins First National Search for Applications of Personal Computing to Aid the Handicapped, New York, N.Y.: IEEE Computer Society Press, 1981.

Rathe, G.H. Computer-assisted instruction and its potential for teaching deaf students. American Annals of the Deaf, 1969, 114 (5), 880-883.

Richardson, Janice E. Computer assisted instruction for the hearing impaired. Volta Review, 1981, 3 (5), p. 328-335.

Russell, John W. The Apple microcomputer in school learning centers. American Annals of the Deaf, 1982, 127 (5), p. 522-528.

Shanahan, Dolores. The computer -- a technology that breaks the sound barrier. American Annals of the Deaf, 1982, 127 (5), p. 476-482.

Slovick, Linda. CAI at CSDF: microcomputer-based authoring systems. American Annals of the Deaf, 1982, 127 (5), p. 536-545.

Watson, Paul G. Utilization of the computer with deaf learners. Educational Technology, 1978, 18 (4), p. 47-49.

Weir, Sylvia, & Watt, Daniel. "LOGO: A computer environment for learning-disabled students." RUN: Computer Education, Monterey, Calif.: Brooks/Cole Publishing Company, 1983, p. 228-238.

Williams, Joseph G. Thorkilolsen, Ron, and Crossman, Edward K. Application of computers to the needs of handicapped persons. RUN: Computer Education, Monterey, Calif.: Brooks/Cole Publishing Company, 1983, p. 228-238.

Wilson, Fred L. Energy and the environment: an approach to teaching modeling that breaks the hearing-impaired student's "limits of growth." American Annals of the Deaf, 1982, 127 (5), p. 645-651.

Withrow, Frank B. Technology and the deaf learner -- preface. Journal of Educational Technology Systems, 1978-79, 7 (1), p. 3-5.

Microcomputer Anxiety

Mr. John Tongue
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

COMPUTER ANXIETY

John Tongue

192-44-5473

The computer is a tool with seemingly limitless possibilities in the field of education. With the advent of the micro-computer a combination of heuristic and algorithmic approaches to the computer can be used on a large scale basis to greatly enhance the quality of education that students receive. But this transformation - this "fourth revolution" - will not come to fruition in the field of education if a passive posture is assumed. For this transformation to occur, questions have to be addressed and barriers have to be overcome. Federal funding is needed for further research, teachers must be retrained, the quality of software must be improved, and administrators must be aware that while the computer is more than a delivery system for drill and practice, it should not be used exclusively in a heuristic fashion. These are just some of the specific problems that must be overcome. But spread among the specific problems are underlying problems that affect the acceptance of the computer and are sometimes overlooked. One such concern is computer anxiety.

What is computer anxiety? What are its causes? What are its effects? Can levels of computer anxiety be determined through testing? Are there methods available for the reduction of computer anxiety? Answers to these questions were sought. Because of the fluid nature of the computing field, books which are often a source of valuable material seemed to have limited information concerning computer anxiety. The majority of useful information that

was found was from research and journal articles. It was interesting to note that the early 1970's saw a flurry of activity in research on computer anxiety which passed by 1975. Just recently further research has been published. This does not imply that computer anxiety was ignored in the interim period. Magazine articles have sporadically referred to computer anxiety, but only a limited number of articles have actually cited findings through research or testing. From those select articles and published research, the information for the paper was gathered. The data can be divided into three basic areas: trait-state anxiety testing, program control/feedback, and terminal experience.

What is computer anxiety? A conceptual distinction has been made in the literature between trait and state anxiety. State anxiety refers to a transitory state or condition that is characterized by feelings of tension and apprehension while trait anxiety implies that an individual is prone to high anxiety levels. Spielberger's State-Trait Anxiety Inventory has been widely used to test computer anxiety. However, even with the distinction between trait and state anxiety, other factors may effect testing. O'Neil (1973) found that worry, and to a lesser degree, emotionality, had a correlation to state anxiety and hypothesized that other uncontrolled variables may be effecting computer anxiety testing. Still, useful research in computer anxiety has come through state-trait anxiety testing.

Rappaport (1971) conducted a study of the interaction between trait anxiety, state anxiety and dogmatism in computer-assisted instruction. While the hypothesized relationship between dogmatism and state anxiety was not confirmed, basic principles of anxiety applied to the computer instruction. High trait anxiety subjects had significantly higher levels of state anxiety than low trait anxiety subjects. A significant interactive effect of math ability and state anxiety was observed. A high anxiety state resulted in more errors for low math ability subjects but had no effect on the performance of high math ability subjects. This apparently was a result of the fact that high math ability students had little difficulty with the material and were not anxious.

O'Neil (1970) devised an experiment which explored the relationship between response mode, trait and state anxiety, and achievement in computer-assisted instruction. One method of instruction was through constructed responses, both long and short, while the other was through reading, which also had long and short forms. As expected, a strong relationship existed between high trait and high state anxiety levels. It was interesting to note that regardless of trait and state anxiety levels, anxiety was higher in the constructed response groups than in the reading groups. (This tendency will be explored later.)

Leherissey (1971) conducted basically the same experiment a year later but added program length as a variable. As O'Neil found, the relationship between trait

and state anxiety was strong and anxiety was higher in constructed response groups. Shortening program length did not reduce anxiety, although it did improve performance.

Other experiments employing state-trait anxiety were employed along the same basic formats. However, as previously mentioned, O'Neil (1973) deduced that perhaps more than aixiety was being tested. He reviewed the literature of Morris and Liebert and wondered if there was a relationship between worry, emotionality, and state anxiety. The study was conducted through a computer-assisted instruction course in introductory educational psychology. While the computer was used as a testing mode, the results that dealt with anxiety were extremely interesting. State anxiety was found to correlate to worry at .6594 while emotionality was correlated at .8037. O'Neil's concern was that much of the work on anxiety experimentation might have been obscured or delimited by related but uncontrolled variables.

The problem of isolating anxiety was encountered by Rohner and Simonson (1982) who developed a Computer Anxiety Index. The test for anxiety consisted of thirty questions, ten of which dealt with computers and twenty distractors which were related to other innovative forms of media. The test was designed to determine the anxiety levels of teachers in respect to computers. The Computer Anxiety Index was found to have a realiability of .88 on the pilot test and .86 on the actual test. However, Rohner and Simonson were not sure that the test was

actually measuring computer anxiety. Through a post hoc analysis they decided that the test was actually measuring computer anxiety and personal preference. These two factors combined related to "intent to use." Like other researchers, Rohner and Simonson suggested that the effort to measure computer anxiety as an independent construct should be continued.

J.B. Hanson (1972) is a researcher who bridged the gap between two of the general categories of computer anxiety research. Hanson was interested in state anxiety but in a different context. His study attempted to determine whether information feedback provided during a difficult task would reduce state anxiety and whether learner control of feedback found lead to further reductions in state anxiety. Hanson found that state anxiety could be reduced through increased feedback. Although feedback generally appears to reduce state anxiety, the learner with high state anxiety is sometimes so anxious that he is unable to effectively use the feedback. Hanson also noted that those students whose programs allowed greater learner control had lower state anxiety levels and improved performance.

O'Neil (Dec. 1972) chose to study a different aspect of feedback; instead of the amount of feedback he concentrated on the type of feedback. The subjects were women college students with extreme scores on the State-Trait Anxiety Inventory. Stress was placed upon the women through feedback in mathematically related learning tasks. The instruction was

presented through an IBM CAI 1500 System. Some people were given negative feedback while others were given rest periods in lieu of the negative feedback. Subjects with high anxiety scores showed a greater increase in anxiety from the pretest than those with low anxiety scores when given negative feedback. O'Neil found no correlation between high anxiety and errors on the computer. However, students who were given rest periods performed better than those who received negative feedback.

One would assume that if feedback has an effect on anxiety levels, then might not the program itself have a great effect on anxiety levels? Hurlock (1974) developed a research project which was in part designed to compare student-controlled computer-assisted instruction to program-controlled computer-assisted instruction. Hurlock studied Navy trainees in Basic Electricity/Electronics School. Hurlock could find no difference in the effectiveness of one program over the other. But perhaps more importantly, the trainees said they liked making "their own decisions about training." Both groups learned as effectively, but the trainees who controlled their programs had a much better attitude. Could this possibly be an example of some of the uncontrolled variables that O'Neil refers to? Perhaps their attitude was better because there was less anxiety involved in student control.

O'Neil experiment (1970) which explored the relationship

between response modes and anxiety is worth reviewing again. There were four types of programs: long constructed response, short constructed response, long reading, and short reading. The experiment was viewed from the state-trait anxiety perspective. Constructed responses were found to evoke higher anxiety levels. But why? Perhaps the subject who was forced to make responses felt that he was not in control of the situation and therefore his anxiety levels rose. If one remembers, Leherissey (1971) conducted the same basic experiment but attempted to reduce anxiety by shortening program length. Although performance was increased, there was no decline in anxiety. The key may well be that the anxiety was not a result of program length but rather lack of control which the students felt when working with a constructed response program.

The final area which has received attention because of its relationship to computer anxiety is terminal experience. Anderson (1981) noted in the Journal of Educational Psychology that after a fifteen to twenty minute orientation program on the Apple II Computer, anxiety levels were significantly reduced. Townsend, (1981) whose article from Technological Horizons in Education entitled Coping Strategies for Resistance to Microcomputers suggested that student's fears and anxieties could be eliminated by gradually employing the computer in the classroom.

Perhaps the most extensive research on the effects of

terminal experience was done by O'Neil (Sept. 1972). While studying anxiety reduction techniques through computer-assisted learning O'Neil investigated the effects of prior terminal experience. He first experimented with a game on the computer and found that it did in fact reduce computer testing anxiety. O'Neil then attempted to give students terminal experience, but in a form more like that of the upcoming testing situation. Instead of a game, a computer-assisted instructional program was used and it was found that anxiety levels were even lower than those which used a game situation. O'Neil hypothesized that the similarity of the program with that of the test was the major factor.

Trait-state anxiety, feedback/learner control and terminal experience - three areas that should be considered when dealing with computer anxiety. King (1975) who has studied attitudes of students and instructors at Lowry Air Force Base noted that as state anxiety scores increase, attitudes towards the computer experience decrease. She suggests that anxiety should be monitored and that a concerted program should be instituted to alleviate potentially stressful aspects of the environment. What can be drawn from the research that has been conducted? First, it is difficult to isolate anxiety and one should realize that other factors may be affecting what one terms "computer anxiety," But even with this limitation, one can still reasonably measure anxiety with the computer because although other factors may have some effect, those factors are

related and need not necessarily be considered detramental. The data from feedback/learner control reveals that an initial computer experience should be one which avoids negative feedback and allows the student to control the program. Terminal experience is also a factor - the greater experience level the greater the reduction in anxiety. But one must remember that the anxiety level will be reduced only if the experience is a positive one. By combining the factors which lead to a reduction in computer anxiety, teachers, administrators, and students should be able to enjoy the benefits of the computer without the limiting effects of anxiety.

BIBLIOGRAPHY

Anderson, Ronald E.; And Others. "The Affective and Cognitive Effects of Microcomputer Based Science Instruction." Journal of Educational Technology Systems, v 9, n 4, 1980-1981, pp. 329-55.

Hanson, Joe B. "An Investigation of Cognitive Abilities, State Anxiety and Performance in a Computer Assisted Instruction Task Under Conditions of No Feedback, Feedback, and Learner Control." AERA Conference, Chicago, Ill., 1972.

Hurlock, Richard E.; And Others. "Student Controlled Versus Program Controlled Computer Assisted Instruction." Naval Personnel and Training Research Lab, San Diego, California, 1974.

King, A[illegible] T. "Impact of Computer-Based Instruction on Attitudes of Students and Instructors: A Review." Technical Training Division, Lowry Air Force Base, Colorado, 1975.

Leherissey, Barbara L.; And Others. "Effects of Anxiety, Response Mode, Subject Matter Familiarity, and Learning Time Achievement in Computer Assisted Learning." Technical Memo Number 42, Florida State University; Talahassee, 1971.

O'Neil, Harold F., Jr. "Anxiety Reduction and Computer-Assisted Learning." Texas University; Austin, Sept. 1972.

-------. "The Effects of Anxiety Reduction Techniques on Anxiety and Computer Assisted Learning and Evaluation of College Students: Final Report. Texas University; Austin, Dec. 1972.

-------. "Effects of State Anxiety and Programming Variables on the Computer-Assisted Learning of College Students; Final Report." Florida State University; Tallahassee, 1970.

-------. "Relationship of Anxiety and Performance in Computer-Assisted Learning." Texas University; Austin. 1973.

Rappaport, Edward. "The Effects of Trait Anxiety and Dogmatism on State Anxiety During Computer Assisted Learning." Florida State University; Talahassee, 1971.

Rohner, Daniel J., and Michael R. Simonson. "Development of an Index of Computer Anxiety." Iowa State University, 1982.

Townsend, Barbara; And Others. "Coping Strategies for Resistance to Microcomputers." Technological Horizons in Education, Nov. 1981, pp. 49-50, 52.

[illegible] (?)

Computer Literacy for Professionls

Mr. Les E. Vance
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

Computer Literacy for Professionals

Leslie E. Vance

March 12, 1983

I had originally intended to make this presentation "Computer Literacy for Professionals" without even attempting to define or describe computer literacy. That decision arose from a sense of frustration: I consider myself computer literate. I've taken several computer programming courses at the college level; I have used computers extensively as research tools; I worked for a short time as a computer programmer. But I have been unable to place my finger on the knowledge that I possess or the abilities that I have that give me the right to call myself computer literate. My frustration was further enhanced by an article in the February 1980 issue of Mathematics Teacher. The article, entitled "Computer Literacy--What is it?" lists some 75 behavioral and informational objectives that the computer literate person should be able to satisfy. Here are a few examples of those objectives:

> Compare computer processing and storage capabilities to the human brain, listing some general similarities and differences.
>
> Select an appropriate attribute for ordering of data for a particular task.
>
> Design an elementary coding system for a given application.
>
> Assess the feasibility of potential applications (Johnson, Anderson, Hansen, and Klassen; 1980 pp.93-94).

There are more than 70 other objectives from six domains: hardware; programming and algorithms; software and data processing; applications; impact; and attitudes, values and motivation. Personally, I would have replaced attitudes, values, and motivation with interactive communication system because of recent hardware developments. It is my feeling that a simple but robust definition of computer literacy is as elusive as a simple but robust definition of literacy.

Harvey Graff, writing in his article "The Legacies of Literacy," indicates that students of literacy have used a number of different measures. . ."ranging from the evidence of written documents, responses to surveys and questionnaires, test results, and the proportions of signatures and marks on documents" (Graff, 1982 p. 13). For one interested in becoming truly literate, these definitions provide little guidance.

Historically, literacy was the domain of scholars and clerics: neither commoner nor lord was lettered. The former had no need of literacy, the latter kept scholars and clerics on staff to handle the chores of reading and writing. The value of universal literacy can be traced to the protestant reformation. It was held that one ought to interpret the scriptures for oneself. Literacy then, was an enabling objective to a terminal objective--i.e., holiness. One approached God alone through one's own understanding of His will based on one's own reading of His Holy Word. Later educational

theorists deemphasized this individualistic purpose of literacy. It came to be viewed as a necessary antecedent to informed and effective participation in a democratic society. Again, literacy is an enabling objective. Consequently, lists of objectives such as the NCTM list are frankensteinian in that regardless of their completeness, they lack a soul, a guiding principle that gives them "life" and direction.

Graff's definitions of literacy (above) are less than satisfactory for two other reasons: (1) They ignore the fact that literacy is not now the ability of a limited few. Literacy is a society-wide context. (2) They provide little guidance for becoming literate. In this presentation, I will focus on this second aspect as it applies to computer literacy. Graff writes: "Literacy, above all, is concerned with the human capability to use a set of techniques. . .their impacts are determined by the manner in which human beings exploit them (Graff, 1982 p. 13)."

I will approach computer literacy as if it were a set of tools: a set of skills and abilities. As such, computer literacy is valuable to the user, not inherently valuable like a diploma, but pragmatically useful because it enables the possessor to perform tasks otherwise impossible or impractical. It is like the chest of tools that a machinist uses in the practice of his trade. Except that the former is abstract, the latter material, the analogy between computer literacy and the machinist's chest of

tools hangs together very well. The analogy may provide guidance for becoming computer literate.

The machinist's tool chest starts out empty as he begins the pursuit of his trade. No two chests contain exactly the same tools: there is no standard set because no two machinists have exactly the same skills or problems to solve. Since the machinist learns to be a machinist by being a machinist, it is unlikely that two will have honed their skills on identical tasks in exactly the same order. Consequently, as a new tool is required, it will be added to the collection of tools already in the chest; this collection of tools will to some extent determine the kinds of new jobs that the machinist tackles. Some of those tools will have been made by the machinist himself because a suitable one was unavailable commercially. Actually, many if not all of his purchased tools represent the solution of an earlier machinist to a problem that he faced. Thus, the machinist's tool box, although fairly unique and individualized, does exist somewhat in the context of a community of machinists. The tools that comprise a machinist's tool box have a sense of beauty about them that may not be obvious to the naive. The machinist uses his tools avocationally in the pursuit of pleasurable ends. The gathering together of the machinist's tolls is a continual process. The tool box and its contents are incredibly expensive--and due to the fact that some of the tools are one of a kind, it is irreplaceable. No one

would question its value to its owner. The IRS recognizes its value to the machinist in the pursuit of his trade by making its acquisition and maintenance tax deductible.

No one these days questions the value of literacy, although that has not always been the case. Computer literacy is apparently following the same path. Some question is value both to the individual and to society but others are unabashed boosters of computer literacy. Among them is the Board of Directors of the National Council of Teachers of Mathematics. The following is an excerpt from a position statement approved by the board in September, 1976.

> An essential outcome of contemporary education is computer literacy. Every student should have first-hand experiences with both the capabilities and limitations of computers through contemporary applications. Although the study of computers is intrinsically valuable, educators should also develop an awareness of the advantages of computers both in interdisciplinary problem solving and as an instructional aid (NCTM, 1976, p. 468).

As you can see, I am barely paying lip service to the criticism of literacy and computer literacy; I have a vested interest in the latter, and take the stand that computer literacy is at least

as valuable to the individual as literacy. This despite the fact that I find it difficult to define computer literacy as a set of terminal objectives. Ignoring for the moment that we are still working without a definition of it, the question at hand is how does one become computer literate? Since we have been comparing computer literacy with literacy, perhaps a continuation of that comparison will be fruitful: the answer to our question may in some way be connected to the answer to the question "how does one become literate?"

At least for the more fortunate among us, while one is yet a child, one goes skipping off to school and learns to read and write in an environment rich in opportunities to read and write. In fact much of the other learning that takes place in school involves reading and writing skills and thus support the development of literacy in the student. One becomes literate--learns to read and write--under the guidance of someone who is not only literate, but also knows a trick or two for facilitating the learning thereof. One hones the skills of literacy by taking part in activities such as the summer reading programs that public libraries sponsor. In other words, to become literate, one reads and writes. I contend that to become computer literate, one computes.

Perhaps someday, computer literacy will receive the widespread institutional support that literacy now does. In fact, many of those who may soon be charged with providing instruction in computer

literacy--professional educators--are themselves only starting out on the path that leads to computer literacy. I would like to facilitate that journey by offering a definition of computer literacy. It is a facetious definition, offered partly in jest, because it does not indicate a destination for such a journey, nor does it offer a travel plan. It is a definition that may motivate the departure and help reduce the expenses of the journey. For the professional educator; computer literacy is. . . tax deductible. Quoting from Internal Revenue Service Publication No. 508, Educational Expenses, "You may deduct expenses for education to maintain or improve skills needed in your job. Ordinarily this includes refresher courses or courses dealing with current developments, as well as academic or vocational courses." Expenses connected with attendance at this conference are tax-deductible.

An implication of this definition of computer literacy is that obtaining it will cost you. It will cost you both in money and time. As mentioned earlier, there are as yet no widespread social institutions for the cultivation of computer literacy. Generally, the only help that you can get from schools and libraries is the use of a book or two about computers. But trying to become computer literate by reading about it is like trying to become literate by talking about it. You will not become literate when, for one reason or another, you can have a book in front of your eyes for only three or four hours a week. By the same token, you will

not become computer literate when, for one reason or another, you can get your hands on an Apple or a TRS-80 for only three or four hours a week.

Continuing with the comparison of computer literacy to literacy, there are no widespread social institutions for the cultivation of computer literacy. If comparable resources are to be invested in the development of your own computer literacy as were invested in the development of your own literacy, those resources will be your own. You will be spending your own money.

If you are to have more than trivial access to a computer (i.e. more than three or four hours per week) then you will buy your own microcomputer. That's not unreasonable; prices for microcomputers now start at under $100.00. . .if you know how to use a soldering iron. Building an operable computer was not one of the objectives on the Johnson, et.al. list; maybe it should have been. But whether you build your own or not, you won't become computer literate by spending a few hours per week at the keyboard and the rest of the time by reading about it or attending classes about it, although that certainly will help. You will become computer literate by computing.

Finagle's eighth rule states: "To study a subject best--understand it thoroughly before you start (Dickson, 1978 p. 58)." Computer literacy is one of those subjects that is covered by the eighth rule. If you know enough about it to know how to go

about becoming computer literate then you already are. That's why I suggest that hands on experience is the best guide for becoming computer literate. If you aren't already computer literate then you won't know which are the right questions to ask. So if you buy a microcomputer, the first question you are likely to ask is: how do I make this thing work? When that question has been answered to your satisfaction, you will already be asking the next question: what is it good for? If you keep track of the questions that you ask and their respective answers, you will be putting together your own answer to the question what is computer literacy? It will very likely be much more satisfying to you than a glib "computer literacy is tax-deductible", or a list of behavioral objectives.

Bibliography

Dickson, Paul. The Official Rules: The Definitive, Annotated Collection of Laws, Principles, and Instructions for Dealing with the Real World. New York: Dell Publishing Co., 1978.

Graff, Harvey J. "The Legacies of Literacy," Journal of Communication, 32, No. 1 (Winter, 1982) pp. 12-26.

Internal Revenue Service. Educational Expenses. Publication No. 508. (Rev. Nov. 81).

Johnson, David C.; Anderson, Ronald E.; Hansen, Thomas P.; and Klassen, Daniel L. "Computer Literacy--What is it?" Mathematics Teacher, 73, No. 2 (Feb. 1980) pp. 91-96.

National Council of Teachers of Mathematics. "Position Statement: Computers in the Classroom," Mathematics Teacher, 71, No. 5 (May 1980) p. 468.

Staff Development for Computer Literacy
A Model Way

Mr. James P. West
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared fo The Second Annual Penn State Microcomputer
Information Exchange Conference

March 11-12, 1983

STAFF DEVELOPMENT FOR COMPUTER LITERACY

A MODEL WAY ?

James P. West

The past 10-12 years has seen a vast array of articles dealing with computers and computer literacy. During the late 1970's, the discussions have focused on the responsibility for building computer literacy among our school students. Molnar (1978) provides a persuasive argument, founded on hard data and rational reasoning , for the necessity of an educational, as well as societal, focus on computer literacy. As the basis for his argument, he perceives an unhealthy rift occurring in society between those who creatively use computers and those who don't. Hirschbuhl (1980) perceives the same occurrence and tells us that if the gap isn't filled, society may have to contend with the highest form of elitism imaginable, a computer elite. Who then is to bring about society's computer literacy? As in the past, the schools are seen as the panacea to eradicate society's ills. The task has been tossed from the Federal Government, to agencies that provide computer services, and back to the schools. Most will agree with Molnar, that if we are to approach equity for all in our educational system and society in general, then the schools must assume responsibility for preparing a computer literate populace.

Papert (1980) sees a different relationship between teacher and student existing with computer-based

microworlds, i.e., constructed realities that allow the human learner to exercise particularly powerful ideas or intellectual skills. The teacher introduces the learner to the microworld in which discoveries will be made, rather than to the discovery itself.

The problem then becomes one of preparing the teachers to use the computer in education. McQuarrie (1982) examined four computer-literacy Staff Development programs and compared them to the Readiness, Planning, Training, Implementation and Maintenance (RPTIM) Staff Development model developed by Wood, Thompson, and Russell (1981).

The purpose of this paper is to examine recent published Staff Development practices designed to prepare teachers to use computers and assess them based on the RPTIM Model. Following the design of McQuarrie (1982), the first section of the paper will describe the RPTIM Model, followed by the description of four computer staff development programs or courses, and finally to compare these staff development practices with the stages of the RPTIM Model.

The RPTIM Model

11 beliefs or assumptions form the basis for the Wood, Thompson and Russell RPTIM Model. These are (1) All personnel in schools, to stay current and effective,

need and should be involved in inservice throughout their careers; (2) Significant improvement in educational practice takes considerable time and is the result of systematic, long-range staff development; (3) Inservice education should have an impact on the quality of the school program; (4) Adult learners are motivated to learn new behaviors when they believe they have control over the learning situation; (5) Educators vary widely in their professional competencies and readiness to learning; (6) Professional growth requires personal commitment to new performance norms; (7) School climate, trust, and peer support for change influence the success of professional development programs; (8) The school is the primary unit of change; (9) The primary responsibility for providing resources and training lies in the school district; (10) The principal is the primary impetus for new program adoption and continued use; (11) Research, theory, and the best education practice are the basis for any effective inservice program (Wood, Thompson, & Russell, 1981).

The stages used to structure the RPTIM Model are Readiness, Planning, Training, Implementation, and Maintenance. Wood and Thompson give the following description of the five stages:

> Stage 1, Readiness, emphasizes selection and understanding of and commitment to new be-

> haviors by a school staff or group of educators. In Stage 11, Planning, specific plans for an inservice program (to be implemented over three to five years) are developed to achieve the desired changes or professional practices selected in Stage 1. In the Training Stage, Stage 111, the plans are translated into practice. The Implementation Stage, Stage 1V, focuses on insuring that the training becomes part of the ongoing professional behavior of teachers and administrators in their own work setting. Stage V, Maintenance, begins as new behaviors are integrated into daily practice. The aim of the final stage is to insure that once a change in performance is operational, it will continue over time (Wood & Thompson, 1980).

The organization of the Stages was developed in a sequential manner so that the Implementation Stage will occur before the Maintenance Stage, etc. For each Stage of the Model, from four to eleven practices have been developed to show the successful implementation of the particular stage.

The Computer Staff Development Programs

As was pointed out earlier in this paper, the amount of literature published on separate disciplines and computer literacy is overwhelming. These unique teachers have been quick to adapt and through personal self-development, have given their students the opportunity to enter the fascinating microworlds of the computer. Individual schools or teachers have been able to successfully use computers in education, but there are few articles that explore the incorporation of district-

wide staff development programs to assist all teachers in becoming computer-literate and learning to use the computer in the classroom. McQuarrie (1982) identified four articles which described staff development programs which have been designed to assist teachers in using computers in the school setting. Four more articles have been identified that also deal with staff development programs or courses and a brief description of each follows.

The Rural School District And the Microcomputer

People who use the computer the most can be valuable resources and should be encouraged to help develop in-service programs and computer use policy. A program of inservice and staff development was instituted in the West Franklin, Kansas school district to familiarize teachers with proper care and use of computers. The sessions were designed to eliminate anxiety and increase staff knowledge.

A combination of handouts, overhead transparencies, and hands-on demonstrations are used to help teachers learn about the computers. After each session, teachers fill out evaluation forms and follow up summaries that are sent to each participant.

The first session, the Introductory Session, is designed to demonstrate the challenge and excitement of

using computers for instructional purposes. It was held in the library with computers set up to demonstrate various CAI programs, a display of software, periodicals containing articles about operation and educational application of the District's microcomputers and operations manuals. It was felt that the session helped reduce teacher apprehension significantly. At the second session, teachers were divided into elementary and secondary levels. This allowed the presenters to provide information and strategies for specific age levels. It also allowed the small groups to have a hands-on experience. The teachers were able to ask questions in a non-threatening atmosphere. The result of this short inservice was that several teachers enrolled in a college credit course in educational application of microcomputers (McDonald & Gibson, 1982).

Computer Literacy for Teachers: High School and University Cooperation

Although computer literacy is an important objective of mathematics instruction, it is not solely the mathematics teachers' responsibility to achieve this goal for students. The mathematics chairpersons from two high schools in the greater New Orleans area diagnosed the problem of computer-illiterate faculty and approached Loyola University for assistance. The result was a co-

operative program to up-grade the computer awareness among the faculty of the high schools. The program that was presented to the faculty of each of the schools consisted of ten hours of lecture, divided into two-hour blocks given at each school on five different nights. There were also many hours of lab work with micro-computers after school hours. The mathematics chairpersons were the moderators for the program. For a week after each lecture, they would answer questions and schedule equipment to insure adequate hands-on experiences. The program of instruction consisted mainly of five discrete steps: (1) Introducti . This session introduced teachers to the micro-computer hardware. The language of micro-computers along with the hierarchy of arithmetic operators, the use of relational and logical operators, and the intrinsic functions can be demonstrated, not to mention some of the basic concepts and commands. Teachers also learn the importa ce of backing up their programs. This first session serves to physically separate the learning of the vocabulary and the grammatical rules of BASIC from the development of an algorithm. (2) Programming. The session begins with the statement of a problem-solving procedure which will be used to write programs. Three programs are written to demonstrate the problem-solving procedure. The programs also serve to demonstrate three

basic structures found in programs,-sequence, selection, and loop. These structures are the basic building blocks of programming and teachers must master these simple ideas before continuing further. (3) Introduction to CAI. The third session begins the introduction to Computer-assisted instruction (CAI). The CAI programs are classified as: drill and practice, tutorials, testing, simulations and dialogue. A program in each category is demonstrated and the philosophy behind the program is discussed. Drill and practice are the most popular programs in education today, but this is not to say that they are the best programs for education. They are relatively easy to write and use. The main point of this session is to help the teacher realize what they can do now and what they can hope to do in the future with more experience. (4) Writing drill and practice programs. By now, the fourth session, teachers are ready to try their own hand at writing drill and practice software. The first program the teachers write is a flash card program followed by a more difficult program that drills the student on the solution to the linear equation ax b = 0. During this session, a discussion on programming tutorials is conducted. For homework, the teachers are asked to write either a drill and practice or a tutorial program in their own discipline. (5) Pro-

gram modification. Too often, programs are purchased blindly and are found to be not exactly what was desired or are realized to have potential in another discipline with "minor" changes. Listing the program reveals multi-statement lines, string functions, and sub-routines that the teachers know nothing about. Now, they learn how to read through a program written by someone else and make cosmetic changes that do not influence the program logic.

The program designers feel that a number of objectives were achieved in a very short time. These objectives were identified as the following: a- each teacher was able to handle the basic operation of the microcomputer system and make a clear distinction between hardware and software; b- each teacher could follow and give correct uotput to a simple flowchart or BASIC program; c- each was able to modify a simple program to accomplish new, but related tasks; d- each learned to detect and correct logic errors in his or her own programs and those of colleagues; e- teachers recognized that computers need instructions to operate and that such instructions come from programs written by people in a programming language; and f- they learned about the types of computer-assisted instruction, availability, and cost (Lopez, 1981).

Who Took a Byte From My Apple ROM?

Twenty-six public and private school teachers spent two weeks of their summer vacation attending a computer summer camp. The objectives of the computer camp were to provide area teachers with an opportunity to learn about microcomputers; and to increase teacher understanding of the role of microcomputers in our society. Twelve Apple 11 plus microcomputers combined with a staff/participant ratio of five to one provided an excellent learning environment for the participants. The camp activities were designed to enable teachers to: evaluate computer software programs; write their own computer-assisted instruction programs for their classes; and develop a lesson plan for teaching the uses of computers. Based on questionnaires and staff interviews, it was found that participants left camp with a feeling of accomplishment and a high level of understanding of the potential of the microcomputer in society and in their classrooms. It was determined that 80 hours of hands-on experience was insufficient to learn more than the basic programming skills. Operation of peripheral devices beyond the use of the printer only proved to frustate most teachers on this initial experience. Burke and Kaiser feel that sending staff back to college or having a local University hold classes in the district

after school is not the answer. They cite the fact that much information and knowledge is lost when a computer course is attended once a week for three months and after school or evening classes do not insure enough hands-on experience as reasons for their opposition. They also fee. one or two-day inservice programs are not the answer since to be successful, participants need to be able to work at the computer for two to four hours at a time. Based on their success, they highly recommend two-week summer camp sessions lasting seven hours per day, including lunch.

Several guidelines for conductin the camp sessions were identified. Among those considered important were; 1- use homogeneous grouping when assigning teachers to sessions; 2- attendance should be open to all teachers in the district, from both public and private schools; 3- pay a stipend each day to cover teacher expenses; 4- attempt to have a good staff/participant ratio; 5- provide a curriculum of vatying resources and much hands-on experience; 6- keep oral presentations to a minimum. Activities that participants felt were the most value to them were those directly related to their use of computers in their classrooms. These included: reviewing and evaluating software; flowcharting; set-up and take-down of machines; writing computer-assisted instruction

programs; and writing programs in BASIC computer language. Those things considered to be of least value to participants included a field trip to a computer center and a lecture-slide presentation on the history and use of microcomputers. The authors feel that you should provide participants with lots of handouts, including terminology and commands, and write programs in BASIC (Burke & Kaiser, 1982).

Computers in the Classroom: An Incentive for Teachers

Ms. Gusè, an assistant principal at Broken Arrow Senior High School, Oklahoma, identified ten steps that introduced her to the microcomputer and how the microcomputer affected her and the curriculum at her school. The first eight steps included; 1) formal training at a NSF Computer Workshop; 2) working with administration in the acquisition of computers; 3) organizing a computer club to test for student interest; 4) developing curriculum for beginning and advanced computer courses; 5) developing in house inservice programs; 6) introducing computers into other curriculum areas and grades; 7) hosting regional computer contests; 8) writing proposals for funding. The ninth step involves offering distinctive computer awareness programs. It is at this point that a staff development program in computer literacy is mentioned. "Computer Literacy for Educators"

is a six-week program that has the following objectives: a- become knowledgeable with computer terminology and the history of computing; b- be able to evaluate software; c- be aware of the many uses of the computer; d- recognize flowcharting as a step in logical thinking, and, e-learn how to read and understand simple BASIC programs. The second course, "Programming in BASIC", runs for eight weeks and lists as objectives: a- understand the problem and plan the solution; b- prepare a flowchart of the problem; c- prepare the instructions for the program in BASIC; and d- test and correct the problem. The tenth and final step is one in which the author and her computer colleagues are acting as resources to neighboring school systems. The staff development program then, involved inservice at Step 5 and computer awareness courses at Step 9 (Gusé, 1982).

Computer Staff Development Practices Contrasted with RPTIM Model

If the RPTIM Model describes effective staff development practices, as the experts indicate it does, then one must agree with McQuarrie (1982) and his findings that current computer staff development programs are weak.

The four programs examined in the literature failed to comply with the Model's Five Stages. However, two of the four cases were incomplete in their information.

These same two, The Rural School District and the Microcomputer, and Computers in the Classroom did partially meet the principles of Stage 1 Readiness in terms of mobilizing support and in terms of the leadership aspect. The Computer Literacy for Teachers and the Computer Summer Camp, although well on their way to viable programs, failed to cite any plans for Stage IV Implementation and Stage V Maintenance. The four to eleven practices as defined by Thompson (1981), are not found in the corresponding stages of the four staff development programs. The following paragraphs will attempt to contrast each of the four programs studied to the RPTIM Model at that Stage.

Readiness: None of the four programs examined goals for the next three to five years. The Computer Literacy Program feels that the teacher can now assess feasibility of a potential educational application and perhaps develop it themselves.

Planning: Although all four programs did meet many practices at this Stage, long range plans were totally in absence. In determining the needs of the staff, again the only option was to participate or not to participate.

Training: All four programs appear to have met the Training practices, with the Summer Computer Camp program appearing excellent in participation. As in the Plan-

ning Stage, all four programs failed to involve all the staff while allowing the training to become a voluntary act.

Implementation and Maintenance: The implementation stage deals with making sure what is learned in the in-service training becomes a part of the activity and behavior of educators in the work setting (Wood, Thompson, and Russell, 1981). None of the programs examined have made clear plans for use in the work setting. The programs also lacked any indication of the maintenance of these skills in computer literacy once acquired.

There appears to be a lack of inservice/staff development programs in the literature that effectively deal with the computer in education. If students are to become computer literate, programs must be instituted that will train and maintain teacher computer literacy in the classroom. If this is not accomplished, we will then truly see Molnar's computer elite in our schools, and, teachers will not be a part of it.

References

Burke, M.A., & Kaiser, J. Who took a byte from my Apple ROM? NASSP Bulletin, 1982, 66 (455), 83-86.

Gose, G.M. Computers in the classroom: An incentive for teachers. NASSP Bulletin, 1982, 66 (455), 12-17.

Hirschbuhl, K. The need for computer literacy and computer applications in the Nation's classrooms. Journal of Educational Technology Systems, 1980, 9 (3), 183-191.

Lopez, A.M. Computer literacy for teachers: High school and University cooperation. Educational Technology, 1981, XXI, (6), 15-18.

McDonald, D. & Gibson, K. A rural school district and the microcomputer. NASSP Bulletin, 1982, 66 (455), 75-77.

McQuarrie, F.O. Staff development for computers in education: A model or a happening. Unpublished report, The Pennsylvania State University, 1982.

Molnar, A.R. The next great crisis in American Education: Computer literacy. Journal of Educational Technology Systems, 1978, 7 (3), 275-283.

Papert, S. Computer-based microworlds as incubators of powerful ideas. In R.P. Taylor (Ed.), The computer in the school: Totor, tool, tutee. New York: The Teacher's College Press, 1980.

Thompson, S.R. A survey and analysis of Pennsylvania public school personnel perceptions of staff devel-

opment practices and beliefs with a view to identifying some critical problems of need. Unpublished doctoral dissertation, The Pennsylvania State University, 1981.

Wood, F.H., Thompson, S.R. & Russell, F. Designing effective staff development programs. Staff Development/Organizational Development, the Association for Supervision and Curriculum Development 1981 Yearbook, Alexandria, VA: Association for Supervision and Curriculum Development, 1981.

The Unexpected Treasure: Developing Problem Solving Skills Through Interactive Fantasy

J. Alan Whiteside
Instructional Systems Graduate Student
The Pennsylvania State University

Prepared for The Second Annual Penn State Microcomputer Information Exchange Conference

March 11-12, 1983

THE UNEXPECTED TREASURE: DEVELOPING PROBLEM SOLVING SKILLS THROUGH INTERACTIVE FANTASY

J. Alan Whiteside

The Pennsylvania State University

Presented to
Second Annual Microcomputer
Information Exchange Conference

University Park, PA.

March 11, 1983

THE UNEXPECTED TREASURE: DEVELOPING PROBLEM SOLVING SKILLS THROUGH INTERACTIVE FANTASY

The commercial tells us that kids don't know whether the cereal is good for them; all they know is that it tastes good. Similarly, those who play computer adventure games may not know that it's good for them; all they know is that it's fun. Strange as the idea may seem at first glance, computer based adventure games can be used to develop problem solving skills. Validation of the instructional usefulness of play and games will lead to a description of the qualities of adventure games that give them the potential for being excellent instructional tools. Given this general background, some characteristics of an ideal instructional problem solving environment will set the stage for showing how adventure games can produce an unexpected treasure--creative problem solving skills--among those who venture into the world of computer based fantasy.

The Instructional Usefulness of Play and Games

The school, the formal learning environment, is usually thought of as being the major source of one's learning. However, a significant portion of one's learning comes not from the direct educational experiences of the school, but from personally interacting with various contexts in play settings. Many educators including Bork (1980) believe there are a number of significant features of play that have instructional usefulness.

Games, and play in general, have a number of inherent characteristics that lend themselves well to accomplishing instructional goals. The nature of play activities allows one to construct a framework of unique, individualized experiences that provide a foundational base for learning. Second, since one really can not play passively, games and play activities provide an opportunity for the "player" to constantly interact with and manipulate the environment. Third, it is to be noted that games, simulations, and simulation games have been shown to be potentially useful in achieving high-level cognitive objectives relating to analysis, synthesis, and evaluation.Games and simulations have found their way into all areas of the educational sphere including business and training applications.

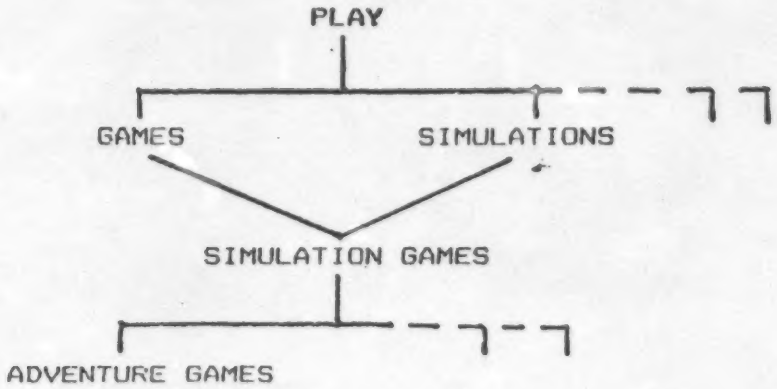
Another very desirable attribute for any mode of instruction is the ability to facilitate transfer of learning. Well-designed games and simulations have been shown to enable participants to apply concepts and principles gained in a gaming situation in other contexts. Since this may be considered the point of all instruction, then the use of games and simulations is worthy of consideration as a viable, instructional method. Finally, games and simulations in an instructional context can potentially foster some desirable process-related skills such as decision-making, which is the basis of problem solving. Such skills influence the way an individual learns to think critically, fluently, and creatively (Gagne and Briggs, 1979).

Characteristics of Adventure Games

Adventure games may be seen as a subset of the category "simulation games" which is a combination of elements from both games and simulations (Figure 1). Games vary widely but all contain similar elements such as players, rules, goals,

competition, choices, and an information system to give the players' status in the game at all times (Dennis, Muiznieks, Stewart, 1979). Simulations can take various forms also, depending on their specific instructional purpose, but their essential characteristic is a representation of a slice of reality with which learners can interact (Bloomer, 1973; Guetzkow, 1963).

Figure 1
Context of Adventure Games



Games and simulations differ in the degree to which they involve the learners. Simulations require learners to actively participate in the learning process because information is only surrendered as a consequence of the learner's interaction. Simulation games, then, are exercises in which players pursue specific goals by making choices appropriate to the context or environment in which they find themselves operating. Their choices must be made as if they were operating within a true real-life environment. Adventure games, which have been called interactive fantasies (Olsen, 1982), may be thought of as a subset of simulation games that do not necessarily have instructional objectives connected with them, but which are played for the sheer enjoyment and challenge they offer. However, because they share the same potentially valuable assets as simulation games, they can conceivably be valid instructional tools. Adventure games portray a slice of potential reality (past, present, or future) or a known true-life situation. Play is governed by a set of rules (which often must be discovered by the players) and the choices made by the players determine progress toward a specified goal. In some instances, updated information about a player's status is "volunteered" by the computer, but at other times it must be requested. Thus, an adventure game presents a competitive environment in which a player must actively pursue a specified goal by using information delivered to him by the computer.

The Problem Solving Environment

If the instructional usefulness of adventure games for promoting problem solving skills is to be determined, it is necessary to know what characteristics an effective problem solving context contains. Gagne & Briggs contend that a well-designed problem solving context requires the learner to invent and use a complex rule to achieve the solution to a problem novel to that individual. In other words, the environment must motivate the

learner toward a goal, whether it be changing a tire without a conventional jack or defeating a sorcerer and collecting his treasure.

Because people have a natural desire to resolve problems and overcome obstacles that prevent their reaching a desired goal, we want to know what motives and procedures lead to a problem's solution so we can use them in other situations. Learners, however, need to be allowed to discover solutions to problems on their own. Stimulating the learner's curiosity by presenting him with problems that are just slightly beyond his grasp at present and by prodding him with questions that point toward a solution leads to independent thinking (Polya, 1957). When a learner is able to think independently, problem solving becomes a much easier task. Stimulating a learner's curiosity, then, is an important characteristic of the problem solving context.

The third element that can promote problem solving skills is a context of challenge. Polya (1962) defines problem solving as "finding a way out of a difficulty, a way around an obstacle, attaining an aim which was not immediately attainable" (p. v). Being faced with a situation that has roused our attention but whose solution is not immediately discernable presents a challenge that appeals to our basic natures as human beings--the desire to solve the problems we encounter.

In order for a learner to acquire some ability in decision-making and in the resultant skill of problem solving, it is necessary to spend time working with problems. Polya (1957) likens the skill to the practical ability to swim. One doesn't learn how to swim by reading a textbook, but rather by practicing and by imitating swimmers. There is no substitute for practice. An effective problem solving environment, therefore, is one in which learners are required to spend time originating, developing, and practicing the skills necessary to solve the problems presented.

Independent thinking does not develop overnight. Watching others solve problems and the experience of solving problems for oneself is invaluable in developing the ability to think independently (Polya, 1957). In addition, when practicing the skill of solving problems, it is valuable to be forced to consider and choose between several approaches to the solution (Polya, 1962). A context in which more than one path to the objective is possible aids in the development of problem solving skills. When only clear-cut answers are expected, the learner's focus will tend to be on how to find the one "right" answer. If, however, a number of alternatives are presented, the learner must develop some strategies (no matter how crude at the outset) to determine the feasibility for success of the alternatives. In this way, discernment and logical thought processes are developed and independent thinking is enhanced.

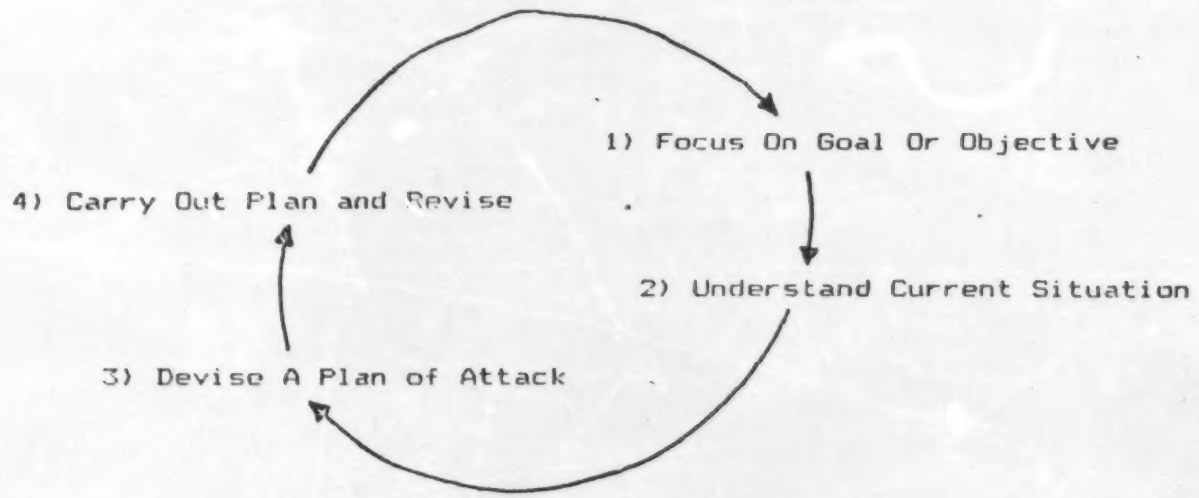
An effective context in which learners can develop problem solving skills should be motivational and challenging. The learner should feel intrinsically compelled to engage himself in the pursuit of the stated goal. His curiosity must be sufficiently stimulated by the problem situation and the alternative solutions so that he is willing to spend a substantial amount of time working with the problem. These elements then lead to the development of the desired problem solving skills.

Developing Problem Solving Skills Through Adventure Games

Given that adventure games embody the instructionally useful qualities of simulation games, the question remains about whether

they contain the elements of effective problem solving environments as set forth above. Another way to understand the elements of an effective problem solving context is to see them subsumed in a general methodology for solving problems. Polya (1957) has advanced such a methodology that outlines various steps through which a learner might proceed. In essence, it may be reduced to four steps: 1) focus on the goal or objective; 2) understand the present situation by comparing the known facts with the unknown; 3) construct a plan for determining the solution; 4) carry out the plan, examine the results, and revise the plan with the goal or objective clearly in mind. This cyclical model is graphically represented in Figure 2.

Figure 2
A Problem Solving Methodology



Many computer based adventure games lend themselves quite well to the origination, development, and practice of problem solving skills using Polya's methodology. The first step is to focus the learner's attention on the objective or the goal. Adventure games typically show (by graphic display) or tell (in a textual form) what objective the player is to reach for. This may be anything from the destruction of evil creatures to the rescue of an entire country. In most instances, the problem is presented as part of a fantasy-world that stimulates the curiosity of the learner as well as challenges him to complete the task before him.

Both graphic adventures (those that show a player's progress through pictures or visual representations of some sort) and the textual fantasies (those that describe the progress of the player verbally) can stimulate creative and imaginative thinking by presenting an initial problem situation or challenge that whets the intellectual and emotional appetite of the player and causes him to actively seek the stated objective. Since it is critical in any instructional context that the learner be actively involved in the process to receive full benefit, adventure games virtually ensure the player's desire to actively pursue attainment of the goal.

As the player begins the game, he must usually assess his current situation by asking the computer for a status report if this information is not supplied immediately. Typical beginning conditions include present location, a list of possessions, and special characteristics of the player such as charisma, strength,

and speed. The player must determine what his assets are in terms of the objective. By making this judgment, the player has begun the process of understanding the known data or the current situation. The point of this step is to connect what is known about the situation with what is not known. The player must analyze what he has to work with and decide what he must do to reach the goal, given the present status.

Once the player understands the problem, he may then begin to determine how to accomplish the goal. Since many adventure games involve conflict with evil characters of various descriptions, this step may be accurately defined as devising a "plan of attack."

In order to construct a workable plan, Polya suggests asking several important questions. Here practical problem solving skills begin to be developed. The learner should ask himself, for example, if a similar problem exists that he has already solved. Studying the known condition is helpful in determining whether a related problem that has already been solved is part of the player's cognitive structure. Because he is able to foresee situations he might face, previously acquired knowledge of some of the game's characteristics increases the player's likelihood of reaching the goal. This is analogous to a person who has repeatedly experienced difficult situations being better equipped to handle the next one encountered. Not only can a player benefit from analyzing his experiences in the current game, but insights gained in prior games are also valuable in determining a plan or devising a strategy for reaching the goal.

The fourth step in the problem solving process is carrying out the plan that has been devised to meet the conditions of the problem. However, when playing most adventure games the first few times, the player usually can not develop an effective, global plan before he is defeated because he lacks so much information that can only be gained through many interactions. Thus, each action is based on a very limited plan devised when the situation is encountered. After the player has suffered defeat several times, he begins to formulate a strategy for actually winning the game. As each stage in the player's plan is successful, it is validated; if he is unsuccessful, he can recalculate and revise the strategy for the next interaction. Actually, defeat is good for teaching how to develop problem solving strategies because the player is forced to reconsider the problem, consolidate his knowledge, and seek a solution in another direction. This enables him to develop the ability to deal with problems outside the adventure game context in a similar manner, and this is an important aspect of the instructional value of adventure games--the learner is able to generalize from a defined setting to larger situations.

Computer based adventure games provide an instructional environment in which problem solving as a particular cognitive strategy can be discovered, used, and developed. In a study to determine what makes computer games fun, Malone (1980) found that three characteristics--challenge, fantasy, and curiosity--make instructional environments more interesting. Adventure games possess all three. In adventure games, then, interest is combined with the elements necessary to support effective problem solving development.

The skills gained by exploring new worlds as one tries to think or fight his way out of a jam should then be transferable from the world of interactive fantasy to real-world problem solving, and a potentially effective instructional medium is born.

REFERENCES

Bloomer, J. What have simulation and gaming got to do with programmed learning and educational technology? PROGRAMMED LEARNING & EDUCATIONAL TECHNOLOGY, 1973, 10:4, 224ff.

Bork, A. Interactive learning. In R. Taylor (Ed.), THE COMPUTER IN THE SCHOOL: TUTOR, TOOL, TUTEE. New York: Teachers College Press, 1980.

Dennis, J. R., & Muiznieks, V. J., & Stewart, J. T. THE ILLINOIS SERIES ON EDUCATIONAL APPLICATION OF COMPUTERS (Vol. 9e). Urbana, Illinois: University of Illinois, 1979.

Gagne, R. M., & Briggs, L. J. PRINCIPLES OF INSTRUCTIONAL DESIGN (2nd ed.). New York: Holt, Rinehart, and Winston, 1979.

Guetzkow, H. SIMULATION IN INTERNATIONAL RELATIONS. Englewood Cliffs, NJ.: Prentice-Hall, 1963.

Malone, T. W. What makes things fun to learn? A study of intrinsically motivating computer games. (Tech. Rep. CIS-7). Palo Alto, CA.: Xerox Palo Alto Research Center, 1980.

Olsen, R. ADVENTURE WRITING. Walled Lake, MI.: Aardvark Technical Services, 1982.

Polya, G. HOW TO SOLVE IT. Garden City, New York: Doubleday Anchor Books, 1957.

Polya, G. MATHEMATICAL DISCOVERY: ON UNDERSTANDING, LEARNING, AND TEACHING PROBLEM SOLVING (2 vols.). New York: John Wiley and Sons, Inc., 1962.